Gaussian Blurring-Deblurring for Improved Image Compression

Moi Hoon Yap[1], Michel Bister[2], Hong Tat Ewe[1]
[1]Multimedia University (MMU)
Jalan Multimedia, 63100 Cyberjaya, Selangor Darul Ehsan, Malaysia
{mhyap, htewe}@mmu.edu.my
http://www.mmu.edu.my
[2]Used to be with Multimedia University; joined University of Nottingham on 01 June 2003
michel.bister@nottingham.edu.my

Abstract. The deblurring of Gaussian blur by inverting the action of the diffusion equation has long been known. This technique is interesting but without much practical application since the images have to be blurred by convolution with a Gaussian to be "de-blur-able" with this technique. In this paper we investigate the possibility to use this blurring-deblurring process as a pre-post-processing step in classical image compression. It is known indeed that the compressibility of an image increases with the blurring, with a relation between *compression ratio* (*CR*) and the blurring scale, *sigma* (σ), which we show to be roughly linear. So, by pre-processing and image with Gaussian blurring before compression, the *CR* will increase. The technique of deblurring Gaussian blur is then used as a post-processing step after decompression. Of course, quantisation of the blurred image prior to deblurring decreases the quality of the deblurring, introducing new errors. Hence, the quantisation step introduced by the compression algorithm also affects the deblurring performed in the post-processing step, resulting in a smaller *Peak Signal to Noise Ratio (PSNR)*. In this article, the complementary effects of increased *CR* and decreased *PSNR* on the *PSNR/CR*-curve of various compression algorithms are studied in function of *sigma*, of the *order* of the deblurring, and of the compression technique.

1 Introduction

A large variety of image compression algorithms have been developed over the years. In this paper we take a special look at one possible approach in image compression, based on the following three premises: 1) smooth images can be compressed much more efficiently than images with a lot of (high-frequency) details; 2) high-frequency details can be removed with appropriate low-pass filtering, but of course this introduces blurring errors; 3) Gaussian blurring can efficiently be de-blurred using the approach in

[1] and [2]. The whole question is: how do the loss of image quality introduced by blurring, the gain in compressibility introduced by blurring, and the gain in image quality introduced by deblurring, mutually relate?

To analyze each of these factors, we will refer to the three different processes mentioned in Figure 1. A number of parameters and choices are to be made: 1) choice of the image; 2) choice of the compression algorithm; 3) *compression ratio (CR), bit rate (bbp)* or *quality factor (Q)* used during compression; 4) amount of blurring (σ_b); 5) order of deblurring (R); 6) sigma value used for calculating the derivatives to do the deblurring (σ_d).

In this paper, we will first examine the clues as to the success of the idea. Next, we will present experimental results highlighting the importance of each of the parameters. Following this, we will discuss the results and present conclusions and clues for future work.

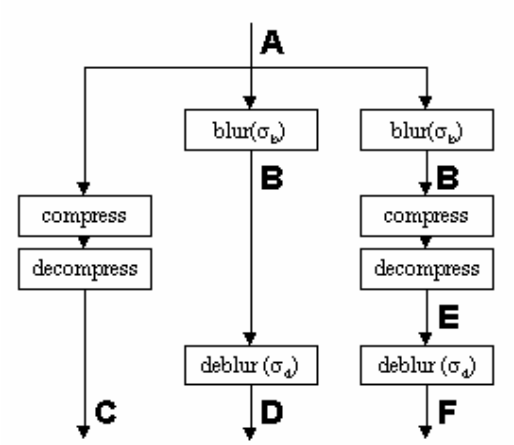


Fig. 1. Starting from an input image **A**, the classical approach to compression is given in the first branch: compression, followed by decompression, resulting in an image **C** which is an approximation of **A**. The techniques introduced in [1] are given in the second branch: Gaussian blurring with variance σ_b, resulting in an image **B**, followed by deblurring, resulting in an image **D**. The approach proposed here is presented in the third branch: pre-process the image to be compressed with a blurring step and post-process with a deblurring step. **E** is the decompressed blurred image, and **F** is the deblurred version of **E**.

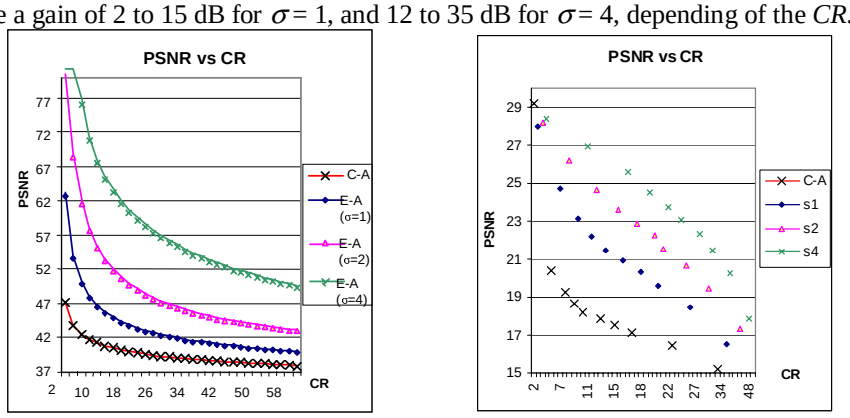
2 Motivation

In this section, we will analyze each of the three premises more in-depth. We will base our discussion on Figure 1, and present some graphs supporting the ideas presented in the premises. In the present paper, we are not necessarily interested in the absolute performance of the compression algorithm, but in the improvement that the proposed pre- and post-processing step (blurring and de-blurring) could bring about. Hence, we chose for two very basic compression schemes. The first one consists of calculating the Discrete Cosine Transform (DCT) of the image and thresholding the coefficients. The ratio of image size to number of non-zero coefficients can be taken as an approximation to the *CR*. The second one is the plain JPEG compression algorithm from Independent

JPEG Group (IJG) [4], with a quality factor *Q* to tune the *CR*. Taking into account that most processing was done in Matlab®, all the images were converted to floating-point representation and rescaled to values between 0 and 1.

First premise: smooth images can be compressed much more efficiently than images with a lot of (high-frequency) details. This makes a lot of sense, since the sharp details in the image produce many high-frequency components in the transformed domain (be it DCT or DWT(Discrete Wavelets Transform)), hence many significant coefficients which all have to be encoded.

Second premise: high-frequency details can be removed with appropriate low-pass filtering, but of course this introduces blurring errors. Simply put, low-pass filtering removes exactly the high frequencies that take up a lot of cost to encode. To illustrate how strong the effect is, we refer to Figure 2(a) for DCT compression, and Figure 2(b) for JPEG compression, where the classical Lena image [5] was blurred with successively larger values of sigma, and then compressed - this in comparison with the compression of the original Lena image. Referring to Figure 1, we compare the compressed blurred image **E** with the blurred image **B**, and for the classical approach the compressed image **C** with the original image **A**. The gain in compressibility is of course dependent on the image and the algorithm used, and on the *CR*. In this case, we notice a gain of 2 to 15 dB for $\sigma = 1$, and 12 to 35 dB for $\sigma = 4$, depending of the *CR*.



(a) (b)

Fig. 2. Improved compressibility of the Lena image as it is being blurred with Gaussian blurring with: **(a)** DCT image compression algorithm, **(b)** JPEG image compression algorithm.

Of course, blurring also degrades the image quality. The severity of this degradation is illustrated in Figure 3, where *PSNR* were calculated between the original and blurred image in function of σ_b. Referring to Figure 1, this means comparing the blurred image **B** with the original image **A**. We notice a loss of 11 to 16dB for σ between 8 and 1.

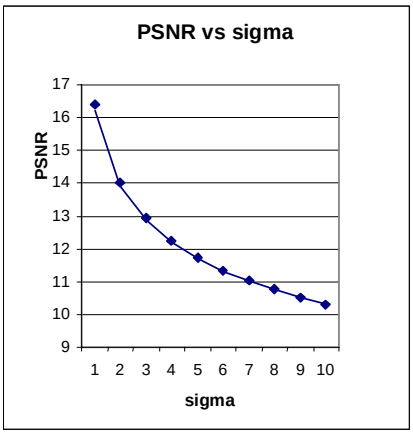


Fig. 3. Loss of quality as a result of Gaussian blurring, in function of the amount of blurring.

Third premise: Gaussian blurring can efficiently be de-blurred. The approach proposed in [1] and [2] is based on a Taylor expansion of the image along the scale axis, and following the expansion in for negative scales, i.e. de-blurring. This Taylor expansion has the following shape:

$$L(x, y; t) = L(x, y; 0) + t\frac{\partial L(x, y; 0)}{\partial t} + \frac{t^2}{2}\frac{\partial^2 L(x, y; 0)}{\partial t^2} + \frac{t^3}{6}\frac{\partial^3 L(x, y; 0)}{\partial t^3} + O(t^4) \quad (1)$$

whereby $L(x, y; t)$ is the image, with x and y the spatial coordinates and t the scaling parameter. The derivatives to the scale can easily be calculated using the diffusion equation:

$$\frac{\partial L}{\partial t} = \frac{\partial^2 L}{\partial x^2} + \frac{\partial^2 L}{\partial y^2} \quad (2)$$

Hence, the scale derivatives can easily be calculated as combinations of spatial derivatives, which, in turn, can be calculated as convolutions of the input image with the derivative of the Gaussian:

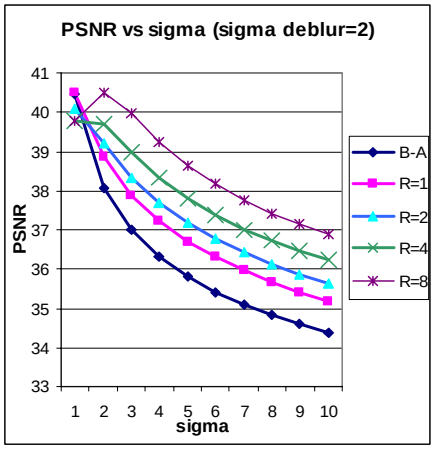
$$\frac{\partial (L \otimes G)}{\partial x} = \Im^{-1}\left(\Im\left(\frac{\partial (L \otimes G)}{\partial x}\right)\right) = \Im^{-1}(\Im(L \otimes G)i\omega) = \Im^{-1}(\Im(L)\Im(G)i\omega)$$
$$= \Im^{-1}(\Im(L)\{\Im(G)i\omega\}) = \Im^{-1}\left(\Im(L)\left\{\Im\left(\frac{\partial G}{\partial x}\right)\right\}\right) = L \otimes \frac{\partial G}{\partial x} \quad (3)$$

The deblurring process depends on two parameters: the amount of blurring used to calculate the derivative operators (which should logically be as small as possible), and the order to which the Taylor expansion is carried. In the case of 3rd order expansion (R = 3), we come to following expression, with derivatives up till the 6th order (from the diffusion equation we notice that an n^{th} order derivative along the scale corresponds to a combination of spatial derivatives of order 2n):

$$L(x, y; t) \approx L(x, y; 0) + t\left(\frac{\partial^2 L(x, y; 0)}{\partial x^2} + \frac{\partial^2 L(x, y; 0)}{\partial y^2}\right) + \frac{t^2}{2}\left(\frac{\partial^4 L(x, y; 0)}{\partial x^4} + 2\frac{\partial^4 L(x, y; 0)}{\partial x^2 \partial y^2} + \frac{\partial^4 L(x, y; 0)}{\partial y^4}\right)$$
$$+ \frac{t^3}{6}\left(\frac{\partial^6 L(x, y; 0)}{\partial x^6} + 3\frac{\partial^6 L(x, y; 0)}{\partial x^4 \partial y^2} + 3\frac{\partial^6 L(x, y; 0)}{\partial x^2 \partial y^4} + \frac{\partial^6 L(x, y; 0)}{\partial y^6}\right) \quad (4)$$

Fig. 5. Comparison of the original Lena image, blurred Lena image (*sigma*=2), deblurred Lena

We now want to evaluate the quality improvement brought about by this type of deblurring, depending on the amount of blurring originally introduced (σ_b) and on the order of the deblurring (R). Referring to Figure 1, this means comparing images **D** and **A**. We see that as the order increases, the quality of the deblurring improves, with roughly +1dB for R=1, +1.5dB for R=2, +2dB for R=4, and +2.5dB for R=8.



Note: R=1 means D-A with order 1, R=2 means D-A with order 2 and so on.

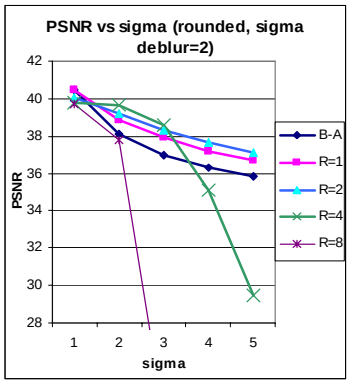
Fig. 4. Improved quality as a result of deblurring Gaussian blur, in function of the amount of blurring, for different values of the order of the deblurring, expressed in *RMSE* (left) and *PSNR* (right).

Putting it all together, it seems like we lose 12dB to 16dB by blurring, but improved compressibility gives us a gain of 2dB to 25 dB, and deblurring can added 1dB to 2.5dB, resulting in a global gain of up to 16dB for σ=4 and R=4, as illustrated in Table 1.

	σ=1		σ=2		σ=4	
	CR=10	*CR*=50	*CR*=10	*CR*=50	*CR*=10	*CR*=50
Blurring	-16dB		-14dB		-12dB	
Compression gain	+5dB	+2dB	+15dB	+5dB	+25dB	+12dB
deblurring	+1dB to +2.5dB					
Expected gain	-10dB to -8.5dB	-13dB to -11.5dB	+2dB to +3.5dB	-8dB to -6.5dB	+14 to +16.5dB	+1dB to +2.5dB

Table 1. Expected lose and gain in PSNR in function of σ, *CR* and *R* image (with order R=1, 2, 4, 8).

However, it is to be noticed in this context that the reconstruction procedure is very sensitive to noise. In particular, since practically all digital images are saved in integer format, the saving of the blurred image before deblurring would result in a rounded-off image **B**, which we will designate as $\hat{}$. The deblurring of such a rounded-off image will be of much poorer quality - even becoming unstable for higher values of σ_b and/or R, as can be seen in Figure 6. In [2], the author wisely remained within the low range of initial blurring, where no instability is noticed until a much higher order (according to his results, until a 32nd order). In this case, referring to Figure 1, images $\hat{}$ and **A** are being compared.



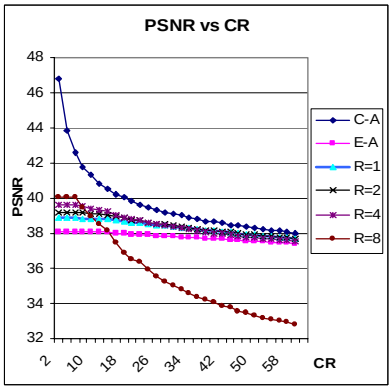
Note: R=1 means ˆ -A with order 1, R=2 means ˆ -A with order 2 and so on.

Fig. 6. Quality of de-blurring Gaussian blur after rounding off the blurred image, in function of the amount of blurring, for different values of the order of the deblurring.

All in all, it seems that one would loose 0.5dB to 2dB in image quality but gain 15 dB in compressibility by blurring the images as pre-processing step to compression, while gaining another 1dB to 2dB by de-blurring the image as a post-processing step. Let us now see how it all fits together.

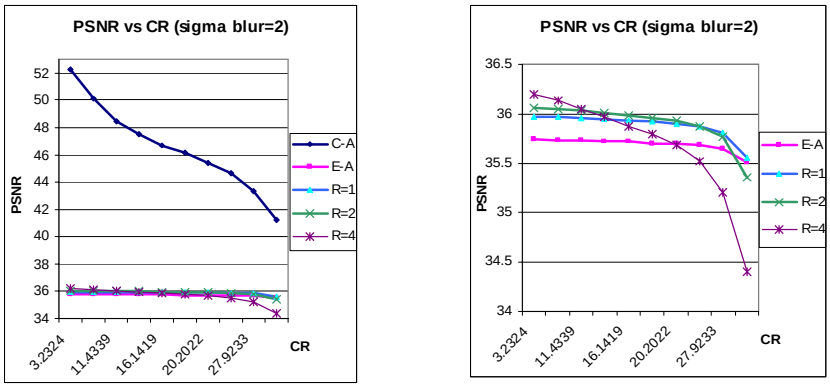
3 Results

Referring to Figure 1, we will now follow the third branch, which is the complete processing: blur the image as pre-processing step, compress-decompress it, and then de-blur it. Figures 7 and 8 show the typical evolution of the *PSNR*: a first plot shows the compression performance without pre- or post-processing (image **C** from Figure 1 compared with image **A**); a second plot with only the pre-processing step (image **E** compared with image **A**); and a third plot with both pre- and post-processing (image **F** compared with image **A**).



Note: R=1 means F-A with order 1, R=2 means F-A with order 2 and so on.

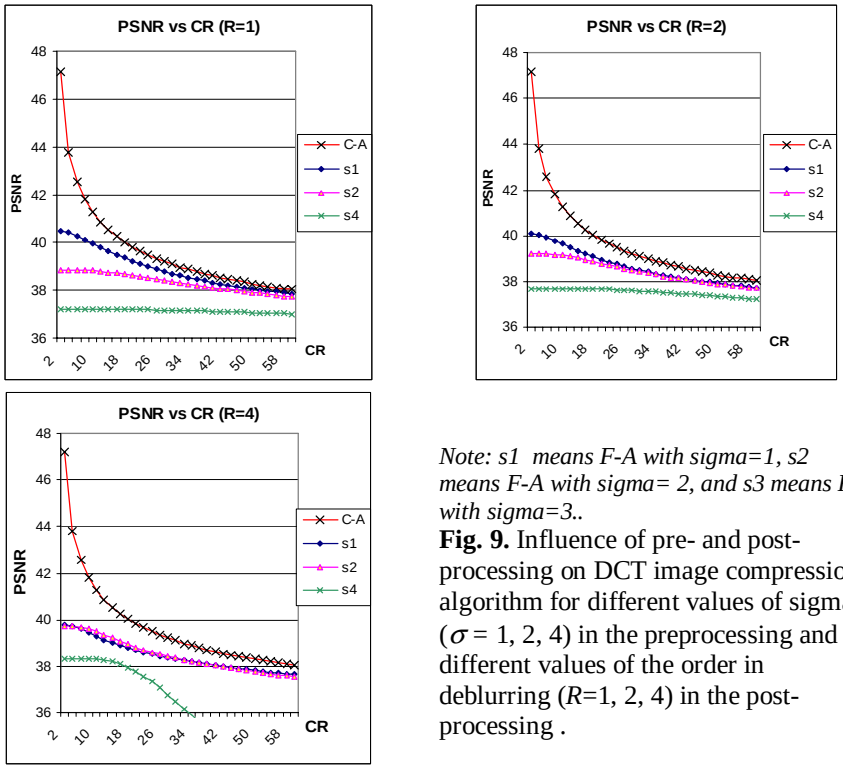
Fig. 7. Influence of pre- and post-processing on the quality of the thresholded DCT as image compression algorithm.



Note: R=1 means F-A with order 1, R=2 means F-A with order 2 and so on.

Fig. 8. Influence of pre- and post-processing on the quality of the JPEG image compression algorithm.

To evaluate the influence of the different parameters on the results, Figures 9 give the results of the modified compression algorithms (with pre- and post-processing, i.e. image **F** compared to **A**) for different values of the initial blur (σ_b) and of the deblurring order (R) for both compression algorithms.



Note: s1 means F-A with sigma=1, s2 means F-A with sigma= 2, and s3 means F-A with sigma=3..

Fig. 9. Influence of pre- and post-processing on DCT image compression algorithm for different values of sigma (σ = 1, 2, 4) in the preprocessing and different values of the order in deblurring (R=1, 2, 4) in the post-processing .

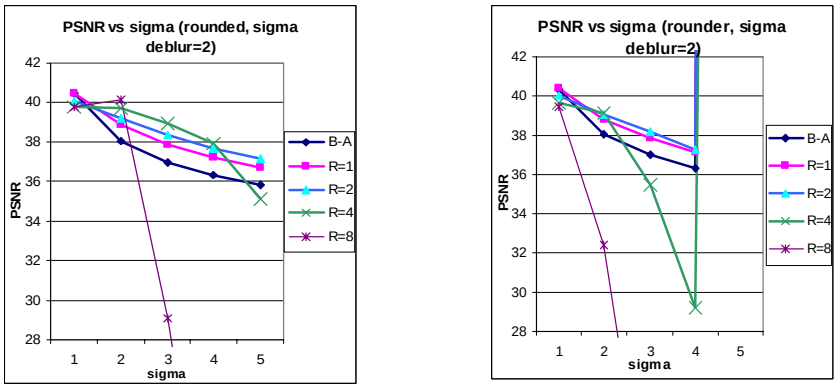
4 Discussion

The results seem to be discouraging: far from improving the quality of the decompressed image, the de-blurring only degrades it further. The explanation for this can be found in Figure 6 and in the JPEG compression algorithm from IJG [4]. The de-blurring process was shown to be sensitive to rounding off of the floating-point values of the blurred image. In our processing chain, however, TWO such quantisation steps can be observed.

First, there is the unavoidable quantisation of the DCT coefficients. However, for low *CR* (high *Q* values), it would be expected that these rounding-off errors should be under control. However, the results remain discouragingly poor.

Second, the JPEG compression algorithm from IJG [4], like most of the compression algorithms available, work with integer input images, since - as noted earlier - most images are saved under integer format. Hence, we are faced with a DOUBLE quantisation step: one that rounds off image **B** to $\hat{}$, and a second one in the compression mechanism itself. This double quantisation is too much and makes the de-blurring algorithm altogether unstable.

To illustrate this point, the experiment of Figure 6 was re-done with coarser and finer quantisations. Figure 6 showed the results of rounding off the values of **B** to 256 possible values before deblurring. Figure 10(a) shows the same process with 1024 output levels, and Figure 10(b) with 64 levels. It is clear that coarser quantisations lead to more instability in the reconstruction.



(a) (b)

Note: R=1 means F-A with order 1, R=2 means F-A with order 2 and so on.

Fig. 10. Quality of de-blurring Gaussian blur after rounding off the blurred image to **(a)** 1024 input values, **(b)** 64 input values in function of the amount of blurring, for different values of the order of the deblurring.

5 Conclusions

It seems a promising idea to improve a compression algorithm by pre-processing it with a Gaussian blurring step so as to improve compression performance, and post-processing it with a Gaussian de-blurring process to reduce the error in the image. However, the double quantisation occurring first at the input of the compression algorithm, next as a necessary step in any lossy compression algorithm, seem to defeat the expected results.

Two possible avenues should still be explored. First, the modification of the compression algorithm so as to take floating point images as input. This would eliminate the spurious quantisation at the entrance of the compression algorithm.

Another possibility would be the use of a lossless compression algorithm. However, any lossless compression algorithm could only work on integer images. Hence, the quantisation step inside the compression algorithm would be avoided, but not the one which round of the blurred image.

By pursuing both avenues, it would be possible to determine which of the two quantisation steps is the most detrimental to the proposed pre- and post-processing step for improved image compression.

References

1. Romeny BMT, Florack LMJ, Salden Ah, Viergever Ma: Higher-order Differential Structure Of Images. Images And Vision Computing, 12(6): 317-325 JUL-AUG (1994).
2. Romeny BMT: Front-End Vision and Multiscale Image Analysis. Kluwer Academic Publishers (2002).
3. John Sporring, Mads Nielsen, Luc Florack and Peter Johansen (Eds.):Gaussian Scale-Space Theory. Kluwer Academic Publishers (1997).
4. [IJG]: http://www.ijp.org (last access: 20 June 2003)
5. Lena images: http://www.geoffdavis.net/wavelet/wavelet.html (last access: 20 January 2003)
6. Scott E Umbaugh: Computer Vison and Image Processing..Prentice-Hall (1998).
7. Luc Florack: Image Structure. Kluwer Academic Publishers (1997).
8. Milan Sonka, Vaclav Hlavac, Roger Boyle: Image Processing Analysis, and Machine Vision. Brooks/Cole Publishing Company (1999).
9. Delores M. Etter: Engineering Problem Solving with Matlab®. Prentice Hall (1997).
10. Alfons H.Salden, Bart M. Ter Haar Romeny, and Max A. Viergever: Linear Scale-Space Theory from Physical Principles. Journal of Mathematical Imaging and Vision, Kluwer Academic Publisher.