Beating the Quality of JPEG 2000 with Anisotropic Diffusion

Christian Schmaltz, Joachim Weickert, and Andrés Bruhn

Mathematical Image Analysis Group, Faculty of Mathematics and Computer Science, Campus, E1.1 Saarland University, 66041 Saarbrücken, Germany
{schmaltz,weickert,bruhn}@mia.uni-saarland.de

Abstract. Although widely used standards such as JPEG and JPEG 2000 exist in the literature, lossy image compression is still a subject of ongoing research. Galić et al. (2008) have shown that compression based on edge-enhancing anisotropic diffusion can outperform JPEG for medium to high compression ratios when the interpolation points are chosen as vertices of an adaptive triangulation. In this paper we demonstrate that it is even possible to beat the quality of the much more advanced JPEG 2000 standard when one uses subdivisions on rectangles and a number of additional optimisations. They include improved entropy coding, brightness rescaling, diffusivity optimisation, and interpolation swapping. Experiments on classical test images are presented that illustrate the potential of our approach.

1 Introduction

Image compression is becoming more and more important due to the increasing amount and resolution of images. Lossless image compression algorithms can only achieve mediocre compression rates compared to lossy compression algorithms, though. Popular lossy image compression algorithms are JPEG [1], which uses a discrete cosine transform, and its successor JPEG 2000 [2], which is based on wavelets.

In the last decade the interpolation qualities of nonlinear partial differential equations (PDEs) have become evident by an axiomatic analysis [3] and by applying them to inpainting problems [4, 5]. Extending inpainting to image compression drives these ideas to the extreme: Only a small subset of pixels is stored, and the remaining image regions are reconstructed by PDE-based interpolation. This idea has been introduced by Galić et al. in 2005 [6] and extended later on in [7]. These authors used edge-enhancing anisotropic diffusion (EED) [8] because of it favourable performance as an interpolation operator. By encoding pixel locations in a B-tree that results from an adaptive triangulation [9] and introducing a number of amendments, they ended up with a codec that outperforms JPEG quality for medium to high compression ratios. However, they could not reach the substantially higher quality of JPEG 2000.

The goal of our paper is to address this problem. We show that it is possible to beat the quality of JPEG 2000 with edge-enhancing anisotropic diffusion,

provided that a number of carefully optimised concepts are combined that have not been considered in [7]: First of all, we replace the adaptive triangulation by a subdivision into rectangles. Afterwards we use an improved entropy encoding of the interpolation data, a rescaling of the brightness map to the interval [0,255], an optimisation of the contrast parameter within the diffusion processes of encoding and decoding, and a swapping of the role of interpolation points and interpolation domain in the decoding step. The resulting novel codec that uses EED within a rectangular subdivision is called R-EED.

Our paper is organised as follows: In Section 2 we review the main ideas behind diffusion-based image interpolation. Our R-EED codec is introduced in detail in Section 3, and its performance is evaluated in Section 4. The paper is concluded with a summary in Section 5.

Related Work. While there are numerous papers that apply nonlinear PDEs and related variational techniques as pre- or postprocessing tools for image and video coding, their embedding *within* the encoding or decoding step has hardly been studied so far. Notable exceptions include work by Chan and Zhou [10] where total variation regularisation is incorporated into wavelet shrinkage, research on specific surface interpolation applications such as digital elevation maps [11], and some recent embedding of inpainting ideas into standard codecs such as JPEG [12].

2 Diffusion-Based Image Compression

As explained in the introduction, the basic idea behind the image compression approach used in this article is to save the brightness values only at a subset $K \subset \Omega$ of the whole image domain $\Omega \subset \mathbb{R}^2$. These values will be denoted by the function $G : K \to \mathbb{R}_0^+$. In order to reconstruct the image, we introduce an artificial time parameter t. The reconstructed version R of the original image I is given by the steady state $R = \inf\limits_{t\to\infty} u(x,t)$ of the evolution equation

$$\partial_t u = Lu \tag{1}$$

with Dirichlet boundary conditions given by G and some elliptic differential operator L. That is, we set the brightness on K to the given values, initialise the remainder of the image arbitrarily, e.g. by setting it to zero, and diffuse the unknown parts of the image until convergence.

As differential operator, we use *edge-enhancing diffusion* (EED) [8] because it has been shown in [7] that it performs favourable in this context. EED is given by

$$Lu = \operatorname{div}(g(\nabla u_\sigma \nabla u_\sigma^\top)\nabla u), \tag{2}$$

where $\nabla u_\sigma := K_\sigma * u$ is the image smoothed with a Gaussian K_σ with standard deviation σ, and g is a diffusivity function. The diffusion tensor $g(\nabla u_\sigma \nabla u_\sigma^\top)$ is a symmetric 2×2 matrix with eigenvectors parallel and orthogonal to ∇u_σ,

and corresponding eigenvalues $g(|\nabla u_\sigma|^2)$ and 1. Here we use the *Charbonnier diffusivity* [13]

$$g(s^2) := \frac{1}{\sqrt{1+\frac{s^2}{\lambda^2}}}, \tag{3}$$

where λ is a *contrast parameter*. Note that EED is designed in such a way that it smoothes along edges, but not across them. Thus, this diffusion process can produce sharp edges.

To compare the performance of different compression algorithms, one considers the *compression ratio*, i.e. the ratio between the file size before and after compression, and some error measure that quantifies the reconstruction error between the initial image I and the reconstruction R. We choose the *mean square error* (MSE)

$$\mathrm{MSE}(I,R) := \frac{1}{|\Omega|}\sum_{\Omega}(R(x)-I(x))^2, \tag{4}$$

since it shows the quality differences between the methods very well. Moreover, there is a monotone mapping from the MSE to the popular peak signal to noise ratio (PSNR).

3 Our Novel Codec

In order to make anisotropic diffusion competitive for image coding great care has to be taken to select an appropriate set of interpolation points and to encode these data in a very compact way. Let us now discuss this in more detail.

3.1 Rectangular Subdivision

The proposed compression algorithm starts by saving the four boundary lines of the image. However, note that whenever we state that we "save" a line of the image, this is done by saving only three points on the line: the two endpoints and the midpoint of the line. Thus, the four boundary lines are saved as eight pixels since pixels lying on several lines must only be saved once.

Next, we check the quality of the image reconstruction when only boundary data are known. Although the complete boundary is not known, this allows to save subimages independently from each other. Then we compute the reconstruction error, i.e. the MSE between the image and the reconstruction. If it is larger than the splitting threshold given by al^d, where a and l are parameters, and d is the recursion depth, the image is split into two subimages by saving a line between the two subimages. These subimages are then saved recursively. The line saved is always the line in the middle of the larger image dimension, as shown in the left image in Figure 1.

Thus, in order to decrease the space required to store the positions of the saved pixels, we do not store points at arbitrary positions, but only save the

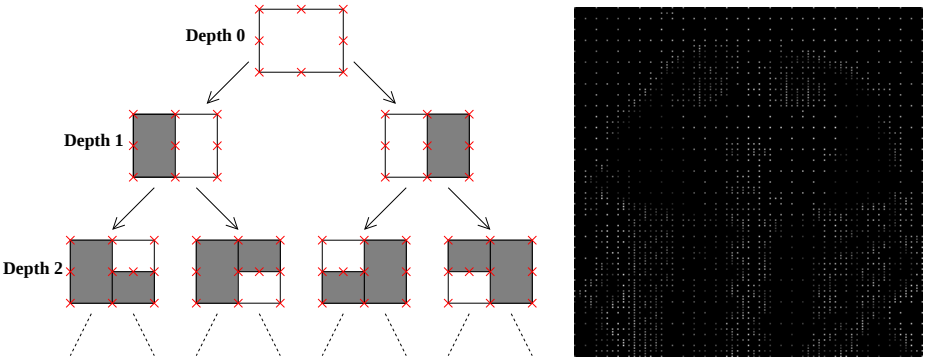

Fig. 1. Left: Illustration of the adaptive grid used for the proposed recursive compression routine. The white area is the area being reconstructed in the corresponding step. **Right**: Example point mask used for compressing the image "trui" with the proposed compression algorithm and a compression ratio close to 60 : 1.

Table 1. The effect of entropy coding for the image "walter", shown in Figure 3, using different splitting thresholds in the subdivision process. Shown are the size of the compressed pixel data in bytes without entropy coding (none), with Huffman coding (HC), with Huffman coding using canonical codes (HCc), with arithmetic coding with static (ACs) or adaptive (ACa) model, Lempel-Ziv-Welch coding (LZW), range coding (RC), gzip (version 1.3.5), bzip2 (version 1.0.3), and PAQ. The best result for each ratio is highlighted.

None	HC	HCc	ACs	ACa	LZW	RC	gzip	bzip2	PAQ
5200	3219	3311	3202	3125	3288	3549	2918	2878	**2366**
2602	1345	1517	1390	1291	1504	1758	1350	1337	**1136**
1270	694	866	716	646	789	1114	683	720	**613**

adaptive grid structure indicated in the left image in Figure 1. This is done by storing the maximal and minimal recursion depth reached in the compression algorithm, as well as one additional bit for each subimage between these two depths. These bits indicate whether the corresponding subimages has been split in the compression step.

3.2 Improved Entropy Coding

To further decrease the file size, the pixel values saved are quantised and coded using a general purpose entropy coder. For the proposed codec, we tested several compression algorithms ranging from Huffman coding [14] over arithmetic coding [15] (with static or adaptive model), Lempel-Ziv-Welch coding [16] to standard tools like gzip (version 1.3.5) and bzip2 (version 1.0.3). Most of the time, PAQ [17] yielded the best results. In our implementation, we used a slightly modified version of paq8o8z-feb28. If very few pixels have to be compressed, a simple arithmetic coding with an adaptive model works best, though. Except for gzip

and bzip2, which are standard tools, and PAQ, the source code of which is available at [18], we used the implementations from [19] in this paper.

The performances of different entropy coders are compared in Table 1 and in Section 4.

3.3 Brightness Rescaling

Some images do not use the whole range of possible grey values. For example, the pixels in the image "trui", which is shown in Figure 2, have a brightness between 56 and 241. Thus, it can make sense to map the brightness of the image such that the complete range is used. This can improve the reconstruction because quantisation has less effects in this way. Note that quantisation does not only occur when quantising the grey values in the encoding step, but also when mapping the real numbers obtained by diffusion to integer values in the decoding.

To illustrate the improvement of this step, we compressed the image "walter" (see Figure 3) once with the method explained in the last section and once with the same method using brightness adjustment. With brightness adjustment, the MSE for a compression rate of approximately 45 : 1 dropped from 50.64 to 46.33.

3.4 Diffusivity Optimisation

In the explanation of EED in the last section, we did not state how to choose λ for the diffusivity in Equation (3). While the same λ was used for all images in [7], we found out that the best parameter is dependent on the image and the compression ratio. Thus, we save the λ parameter that should be used in the reconstruction. We assume this parameter is between 0 and 1, quantise it to 256 different values and use a single byte to store it.

Furthermore, we noticed that a different λ parameter should be used in the compression and decompression steps. This is advantageous due to two reasons: Firstly, the subimages reconstructed in the compression step necessary to generate the grid structure are not equal to the corresponding subimages in the reconstructed image since the influence of surrounding parts of the image are neglected. Secondly, the compression algorithm raises or lowers the saved brightness of each saved point if that improves the reconstruction error, similar to the approach proposed in [7]. During these point optimisations, the optimal λ for the reconstruction may change. Thus, searching an optimal λ and performing the point optimisations is interleaved. That is, after the optimal λ is found, each saved point is optimised once, and these two steps are repeated until convergence.

When using an optimised λ in the compression step, the MSE of our test image "walter" improves from 46.33 to 38.91. After using the optimised parameter for the decompression, we get an error of 38.38. Using one point optimisation step, the error drops to 24.67, and finally to 21.38 after multiple optimisations.

3.5 Interpolation Swapping

Due to quantisation, most points stored in the compressed file are actually slightly inaccurate. This effect can be even stronger after the point optimisations explained in the last section.

To ease this problem, we follow an idea by Bae [20] and perform an additional step after the decompression step explained so far: Once the image is reconstructed, we swap the roles of known and unknown points. That is, the points reconstructed by diffusion are assumed to be known, and the previously known points on the interpolation mask are reconstructed with EED. With this interpolation swapping step, the reconstruction error of the image "walter" drops from 21.38 to 20.13.

We abbreviate our EED-based image compression method with rectangular subdivision, improved entropy encoding, brightness rescaling, and diffusivity optimisation by R-EED.

3.6 File Format

The image format used by the proposed algorithm is given by:

- image size (between 8 (small image, equal width and height) and 18 bits)
- entropy coder used (4 bits, see Section 3.2)
- brightness mapping information (between 1 and 17 bits, see Section 3.3)
- contrast parameter for decompression (1 byte, see Section 3.4)
- flag if interpolation swapping should be used (1 bit, see Section 3.5)
- minimal and maximal recursion depth (2 bytes, see Section 3.1)
- splitting information (variable size, see Section 3.1)
- compressed pixel data (variable size)

In our implementation, there are currently four additional bits used for flags which had been used to test extensions not used any more. Thus, these bits can be used for checksums.

4 Experiments

In this section, we show compression results of the proposed algorithm for different compression rates and compare the results against JPEG, JPEG 2000, and the approach from Galić et al. [7]. For all experiments, we set σ to 0.8.

The first image for which results are presented is the image "trui". This image is a standard test image often used in image processing. In order to compare our algorithm against the one proposed in [7], we scaled the image to 257×257, since that resolution was used there. Figure 2 shows the result using the different compression methods. The images for JPEG and JPEG 2000 have been created with "convert" (version ImageMagick 6.2.4 02/10/07). Note that convert uses optimised entropy coding parameters when saving JPEG files. Since this was not the case for the JPEG images shown in [7], those images are worse than the JPEG images shown here.

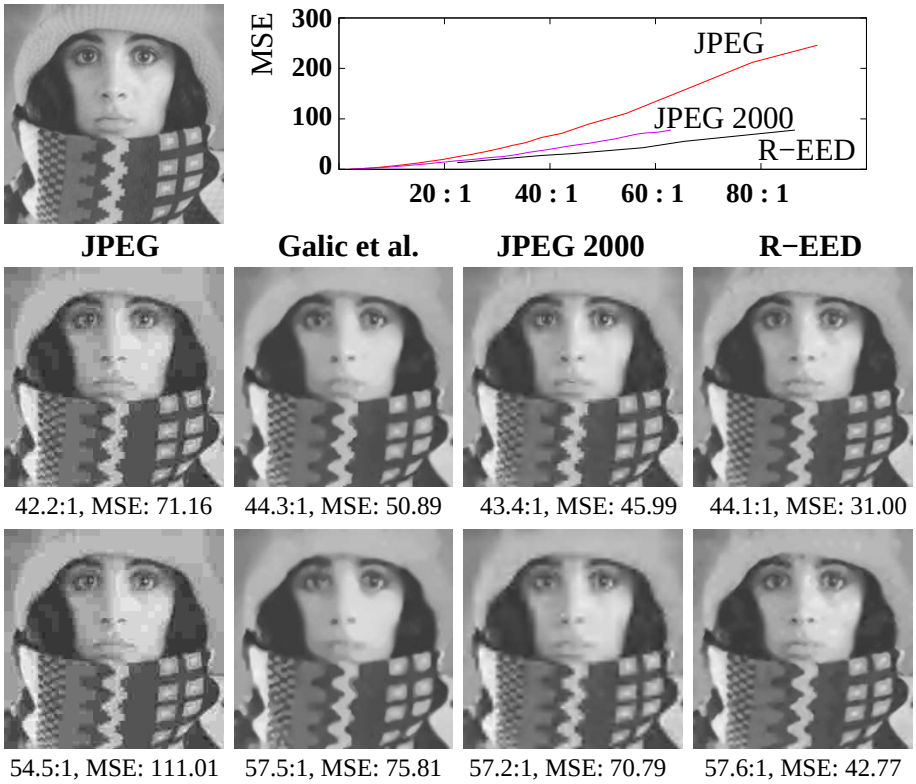

Fig. 2. Comparison of the image quality when saving the image "trui", scaled to 257×257 with different compression algorithms. **Top row**: Input image, and plots showing the MSEs for different compression ratios. **Middle row**: Images obtained with JPEG, the method by Galić et al. [7], JPEG 2000 and with the proposed method (R-EED) with a compression rate close to $43 : 1$. **Bottom row**: Results with compression rate close to $57 : 1$. The images showing the compression result of [7] are courtesy of Irena Galić.

Nevertheless, JPEG is clearly worse than the other approaches, though. In order to demonstrate the performance difference between the proposed approach and JPEG 2000, we compute their MSE for comparable compression ratios. We observe that the reconstruction error for JPEG 2000 is substantially inferior to R-EED: For compression ratios around $43 : 1$, the JPEG 2000 error is 48 % larger, and for ratios around $57 : 1$ it is even 66 % worse.

Compared to the approach by Galić et al. [7], R-EED also yield clearly superior results. The images created by Irena Galić, which are shown in Figure 2, have an MSE of 50.89 (compression ratio $44 : 1$) and 75.81 (compression ratio $58 : 1$), which is 64 % and 77 % higher than that obtained with R-EED.

Furthermore, let us also consider two more images: the first image of an image sequence of Walter Cronkite, available at [21], and a subimage of the

Table 2. Compression results for JPEG, JPEG 2000, and for the proposed algorithm (R-EED) for the images "trui", "walter", and a subimage of "peppers". The best results are highlighted.

	trui		**walter**		**peppers**	
	Ratio	MSE	Ratio	MSE	Ratio	MSE
JPEG	42.17 : 1	71.16	45.15 : 1	39.67	42.03 : 1	70.47
JPEG 2000	43.44 : 1	45.99	45.40 : 1	27.55	42.57 : 1	48.97
R-EED	44.11 : 1	**31.00**	45.40 : 1	**20.13**	42.96 : 1	**42.61**

image "peppers". For both images, the proposed compression algorithm beats JPEG 2000 for high compression rates, and achieves a similar performance for medium compression rates, as is demonstrated in Figure 3 and Table 2.

The graphs in Figure 3 also show results of the proposed algorithm when only Huffman coding is used. As can be seen, even with this simple entropy coder, we still achieve a better quality than JPEG 2000 for high compression ratios.

Note that the graphs in the figures show the complete compression range for JPEG and JPEG 2000, i.e. with quality settings from 100 to 1. However, only a small subinterval of the results obtainable with the proposed algorithm is shown.

Results in which the image "trui" was compressed with compression ratios up to 186.77 : 1 are shown in Figure 4. Higher compression ratios are also possible.

5 Conclusions

We have presented an image compression method that performs edge-enhancing anisotropic diffusion inpainting on an adaptive rectangular grid. By using an improved entropy coding step, brightness rescaling, an optimised diffusion parameter in the compression as well as in the decompression step, and interpolation swapping, the proposed algorithm can yield results that clearly surpass those of related previous work [7] as well as of JPEG and even the sophisticated JPEG 2000 standard.

Our ongoing work includes research on parallelisation strategies for multicore architectures, optimal handling of highly textured regions, as well as extensions to colour images and videos.

Acknowledgement. We thank Irena Galić for fruitful discussions and for providing two images in Figure 2.

References

1. Pennebaker, W.B., Mitchell, J.L.: JPEG: Still Image Data Compression Standard. Springer, New York (1992)
2. Taubman, D., Marcellin, M.: JPEG2000: Image Compression Fundamentals, Practice and Standards. Kluwer Academic Publishers (2002)
3. Caselles, V., Morel, J.M., Sbert, C.: An axiomatic approach to image interpolation. IEEE Transactions on Image Processing **7**(3) (March 1998) 376–386

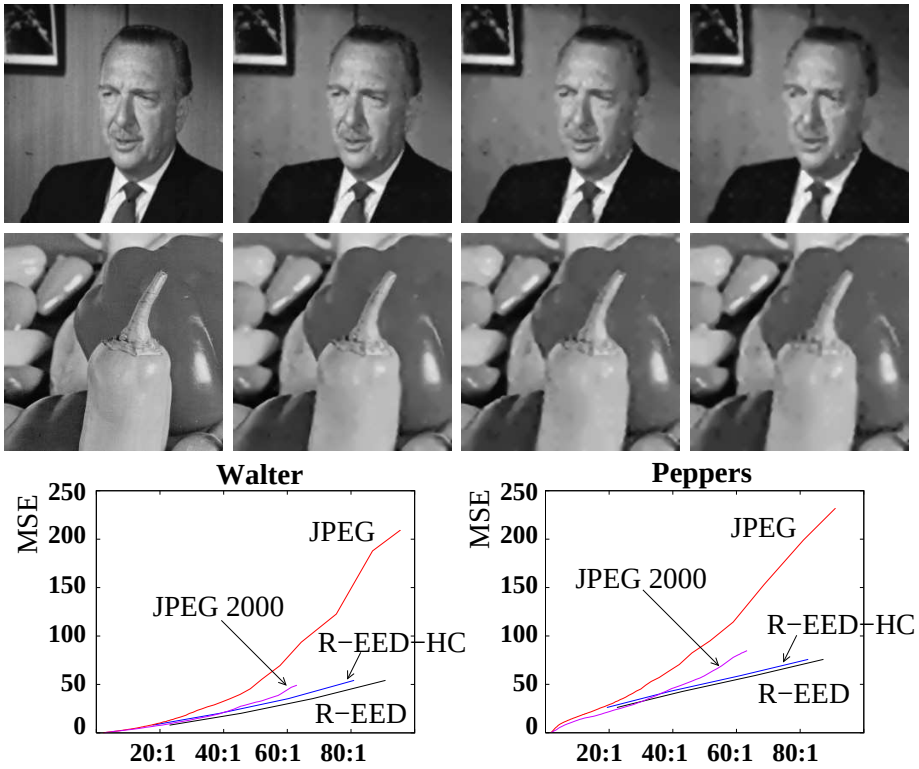

Fig. 3. Compression results with the images "walter" (with resolution 256×256) and a 257×257 subimage of the image "peppers". Shown are, from left to right, the initial image, and results obtained with our approach for compression rates of approximately $44 : 1$, $66 : 1$, and $89 : 1$. The graphs show the performance of the proposed algorithm with the best entropy coder (R-EED) or with Huffman coding (R-EED-HC), and of JPEG and JPEG 2000 for various compression rates of these images.

Fig. 4. Experiments with different compression ratios for the image "trui", scaled to 257×257. **From left to right**: Compression ratio of $86.43 : 1$, $135.76 : 1$, and $186.77 : 1$.

4. Masnou, S., Morel, J.M.: Level lines based disocclusion. In: Proc. 1998 IEEE International Conference on Image Processing. Volume 3., Chicago, IL (October 1998) 259–263
5. Bertalmío, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proc. SIGGRAPH 2000, New Orleans, LI (July 2000) 417–424
6. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.P.: Towards PDE-based image compression. In Paragios, N., Faugeras, O., Chan, T., Schnörr, C., eds.: Variational, Geometric and Level-Set Methods in Computer Vision. Volume 3752 of Lecture Notes in Computer Science. Springer, Berlin (2005) 37–48
7. Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaev, A., Seidel, H.P.: Image compression with anisotropic diffusion. Journal of Mathematical Imaging and Vision **31**(2–3) (July 2008) 255–269
8. Weickert, J.: Theoretical foundations of anisotropic diffusion in image processing. Computing Supplement **11** (1996) 221–236
9. Distasi, R., Nappi, M., Vitulano, S.: Image compression by B-tree triangular coding. IEEE Transactions on Communications **45**(9) (September 1997) 1095–1100
10. Chan, T.F., Zhou, H.M.: Total variation improved wavelet thresholding in image compression. In: Proc. Seventh International Conference on Image Processing. Volume II., Vancouver, Canada (September 2000) 391–394
11. Solé, A., Caselles, V., Sapiro, G., Arandiga, F.: Morse description and geometric encoding of digital elevation maps. IEEE Transactions on Image Processing **13**(9) (September 2004) 1245–1262
12. Liu, D., Sun, X., Wu, F., Li, S., Zhang, Y.Q.: Image compression with edge-based inpainting. IEEE Transactions on Circuits, Systems and Video Technology **17**(10) (October 2007) 1273–1286
13. Charbonnier, P., Blanc-Féraud, L., Aubert, G., Barlaud, M.: Deterministic edge-preserving regularization in computed imaging. IEEE Transactions on Image Processing **6**(2) (1997) 298–311
14. Huffman, D.A.: A method for the construction of minimum redundancy codes. Proceedings of the IRE **40** (1952) 1098–1101
15. Rissanen, J.J.: Generalized Kraft inequality and arithmetic coding. IBM Journal of Research and Development **20**(3) (1976) 198–203
16. Welch, T.A.: A technique for high-performance data compression. Computer **17**(6) (1984) 8–19
17. Mahoney, M.: Adaptive weighing of context models for lossless data compression. Technical Report CS-2005-16, Florida Institute of Technology, Melbourne, Florida (December 2005)
18. Mahoney, M.: Data compression programs. http://www.cs.fit.edu/~mmahoney/compression/ . Last visited March 01, 2009.
19. Dipperstein, M.: Michael dipperstein's page o'stuff, homepage. Available from http://michael.dipperstein.com/index.html . Last visited January 22, 2009.
20. Bae, E., Weickert, J.: Partial differential equations for interpolation and compression of surfaces. In: Proc. Seventh International Conference on Mathematical Methods for Curves and Surfaces. Lecture Notes in Computer Science, Berlin, Springer (2008) To appear.
21. Signal and Image Processing Institute of the University of Southern California: The USC-SIPI image database. http://sipi.usc.edu/database/index.html. Last visited March 01, 2009.