In T. Bräunl, B. McCane, M. Rivera, X. Yu (Eds.):
Image and Video Technology.
Lecture Notes in Computer Science, Vol. 9431, 63-74, Springer, Cham, 2016.
The final publication is available at link.springer.com

# From Optimised Inpainting with Linear PDEs towards Competitive Image Compression Codecs

Pascal Peter[1], Sebastian Hoffmann[1], Frank Nedwed[1],
Laurent Hoeltgen[2], and Joachim Weickert[1]

[1] Mathematical Image Analysis Group, Faculty of Mathematics and Computer Science, Campus E1.7, Saarland University, 66041 Saarbrücken, Germany.
{peter,hoffmann,nedwed,weickert}@mia.uni-saarland.de
[2] Applied Mathematics and Computer Vision Group, Brandenburg University of Technology, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany
hoeltgen@b-tu.de

**Abstract.** For inpainting with linear partial differential equations (PDEs) such as homogeneous or biharmonic diffusion, sophisticated data optimisation strategies have been found recently. These allow high-quality reconstructions from sparse known data. While they have been explicitly developed with compression in mind, they have not entered actual codecs so far: Storing these optimised data efficiently is a nontrivial task. Since this step is essential for any competetive codec, we propose two new compression frameworks for linear PDEs: Efficient storage of pixel locations obtained from an optimal control approach, and a stochastic strategy for a locally adaptive, tree-based grid. Suprisingly, our experiments show that homogeneous diffusion inpainting can surpass its often favoured biharmonic counterpart in compression. Last but not least, we demonstrate that our linear approach is able to beat both JPEG2000 and the nonlinear state-of-the-art in PDE-based image compression.

**Keywords:** linear diffusion inpainting, homogeneous, biharmonic, image compression, probabilistic tree-densification

## 1 Introduction

For a given image, the core task of image compression is to store a small amount of data from which the original can be reconstructed with high accuracy. Homogeneous and biharmonic partial differential equations (PDEs) can restore images with a high quality from a small fraction of prescribed image points, if their position and value are carefully optimised [1,2,6,15]. This suggests the viability of linear, parameter-free PDEs for image compression. Unfortunately, all these data optimisation efforts are in vain, as long as one cannot store the data efficiently.

Currently, PDE-based compression is dominated by nonlinear models that require parameter optimisation, e.g. edge-enhancing anisotropic diffusion (EED) [25]. In particular, the general purpose codec R-EED by Schmaltz et al. [23] can beat popular transform-based coders like JPEG [18] and JPEG2000 [24]. In contrast, homogeneous diffusion has only been applied for specialised applications (e.g. cartoon compression in [16]) which do not employ the sophisticated data optimisation methods mentioned above. Moreover, there is no compression codec based on biharmonic inpainting at all. This shows the lack of general purpose codecs with linear inpainting PDEs.

**Our contribution.** To fill this gap, we investigate how to embed powerful data optimisation strategies for harmonic and biharmonic inpainting into two codecs: 1. A compression scheme that combines free choice of known pixels by optimal control [2, 6] with tailor-made entropy coding. 2. A stochastic method that restricts pixel selection to a locally adaptive grid, but allows to store these positions efficiently as a binary tree. We evaluate how individual restrictions and lossy compression steps of both frameworks affect the performance of harmonic and biharmonic inpainting. In addition, we compare our best methods against the state-of-the-art in PDE-based compression and the quasi-standards in transform-based compression.

**Related work.** Homogeneous diffusion has been applied for the compression of specific classes of images. In particular, Mainberger et al. [14] have proposed a highly efficient codec for cartoon images, and there are several successful coders for depth maps [5, 7, 12]. However, unlike our approach, these methods rely primarily on semantical image features such as edges. This choice is motivated by the theoretical results of Belhachmi et al. [1] which suggest to choose known data at locations with large Laplacian magnitude. Köstler et al. [11] apply homogeneous diffusion for real-time video playback on a Playstation 3.

General purpose codecs with PDEs are mainly based on edge-enhancing anisotropic diffusion (EED) [25] and efficient representations of data locations by binary trees. Initially, this class of methods was proposed by Galić et al. [4], while the current state-of-the-art is the R-EED codec by Schmaltz et al. [23]. Modifications and extensions of R-EED include colour codecs [19], 3-D data compression [23], and progressive modes [22].

In addition, there are several works that are closely related to compression, but do not consider actual encoding [1, 2, 6, 15]. Instead, they deal with optimal reconstruction from small fractions of given data. We directly use results from the optimal control scheme for harmonic PDEs by Hoeltgen et al. [6] and its biharmonic extension. Our densification approach on restricted point sets is inspired by the approach of Mainberger et al. [15]. They consider a stochastic sparsification on unrestricted point sets.

**Organisation of the paper.** We begin with a short introduction to PDE-based inpainting and the optimisation of spatial and tonal data in Section 2. Section 3 proposes solutions to general challenges of PDE-based compression algorithms. These form the foundation for our codecs. In Section 4 we describe a new general compression approach for exact masks, and we apply it to the optimal control method of Hoeltgen et al. [6]. Section 5 introduces a stochastic, tree-based method that imposes restrictions on the location of known data. We evaluate both compression frameworks in Section 6. Section 7 concludes our paper with a summary and outlook on future work.

## 2 PDE-based Inpainting

**Image Reconstruction.** In PDE-based image compression, we want to store only a small fraction of the original image data and reconstruct the missing image parts. To this end, consider the original image $f : \Omega \to \mathbb{R}$. It maps each point from the rectangular image domain $\Omega$ to its grey value. Let us assume that the set of known locations $K \subset \Omega$, the *inpainting mask*, is already given and we want to reconstruct the missing data in the *inpainting domain* $\Omega \setminus K$.

For a suitable differential operator $L$, we obtain the missing image parts $u$ as the steady state for $t \to \infty$ of the evolution that is described by the PDE

$$\partial_t u = Lu \quad \text{on } \Omega \setminus K. \tag{1}$$

Here, we impose reflecting boundary conditions at the image boundary $\partial\Omega$. In addition, the known data is fixed on $K$, thus creating Dirichlet boundary conditions $u = f$. In our work, we consider two different parameter-free choices for the differential operator $L$. In the simplest case, we apply *homogeneous diffusion* [9]: $Lu = \Delta u = \mathrm{div}(\boldsymbol{\nabla} u)$. Since experiments suggest that the *biharmonic operator* $Lu = -\Delta^2 u$ may give better reconstructions [2,4,23], it is also considered.

Both operators propagate known information equally in all directions and behave consistently throughout the whole image evolution. In contrast, more complex models in compression are nonlinear and anisotropic: edge-enhancing diffusion (EED) [25] inhibits diffusion across image edges. In principle, this allows EED to obtain more accurate reconstructions from the same amount of known data [4,23]. However, the price for this increase in quality are algorithms with higher computational complexity and the need for parameter optimisation. Note that the compression frameworks that we propose in the following Sections work in a discrete setting. To this end we consider the finite difference approximations of the inpainting equation in the same way as in [15].

## 3 From Inpainting to Compression

**Spatial and Tonal Optimisation.** For a pure inpainting problem, predetermined missing image regions are reconstructed from known data. This is fundamentally different in compression, where the whole image is known and the encoder actively chooses known pixels. Many research results have confirmed that the choice of known data influences the reconstruction significantly [1,2,6,15]. At the same mask density, i.e. the same amount of known data, choosing optimal

```
Input: Original image $\boldsymbol{f}$, admissible set of quantised grey values $Q := \{q_1, ..., q_n\}$,
    inpainting mask $\boldsymbol{c}$.
Initialisation: $\boldsymbol{u} := r\,(\boldsymbol{c}, \boldsymbol{f})$ and $\boldsymbol{g} := \boldsymbol{f}$.
Compute:
    For all $i \in K$:
        Compute the inpainting echo $\boldsymbol{b_i}$.
    Do
        For all $i \in K$:
          1. Compute the correction term $\alpha := \frac{\boldsymbol{b_i}^\top (\boldsymbol{f}-\boldsymbol{u})}{|\boldsymbol{b_i}|^2}$.
          2. Set $\boldsymbol{u}^{\text{old}} := \boldsymbol{u}$.
          3. Update the grey value $g_i := g_i + \alpha$.
          4. Apply coarse quantisation: $g_i' := \text{argmin}_{q \in Q} |g_i - q|$
          5. Update reconstruction $\boldsymbol{u} := \boldsymbol{u} + \alpha' \cdot \boldsymbol{b_i}$ with $\alpha' = g_i' - u_i$.
    while $|\text{MSE}(\boldsymbol{u}, \boldsymbol{f}) - \text{MSE}(\boldsymbol{u}^{\text{old}}, \boldsymbol{f})| > \varepsilon$.
Output: Optimised quantised grey values $\boldsymbol{g}$.
```

Algorithm 1: Quantisation-aware grey value optimisation.

positions and pixel values is vital for a good reconstruction. In our paper we consider a separate optimisation process of both types of data. First we optimise the locations of data and then perform a grey value optimisation (GVO) step.

**Efficient Storage.** In PDE-based image compression, the reconstruction capability of the inpainting operator is only one of two important factors for the success of a codec. The other key element is the efficient storage of known data. There is a straightforward trade-off between a desired sparsity of stored points and reconstruction quality. In addition, a codec can influence the final file size by combining multiple other lossless and lossy compression steps. In the following, we discuss different possibilities for such compression steps and how they can be combined with the aforementioned spatial and tonal optimisation.

**Entropy Coding** is an essential concept of compression that is employed in most successful codecs. It aims at removing redundancy from data and thereby stores it losslessly, but with a reduced file size. Huffman coding [8], adaptive arithmetic coding [21] and PAQ [13] have been successfully used in PDE-based compression [23]. In our setting there are always two different kinds of known data that must be stored: grey values and locations on the pixel grid. In addition, some header information like file sizes or parameters need to be stored. Therefore, we choose to use PAQ for our codecs, since it is a context mixing scheme that can locally adapt to different types of data. PAQ combines models that predict the next bit in a file from a history of already compressed bits. With neural networks, it adapts the weighting of these models to changing content types throughout a single file. Thus, it can be directly applied as an efficient container format for both positional and brightness data.

**Quantisation.** For brightness data, the performance of the aforementioned entropy coders can be improved by a coarser quantisation. Instead of storing grey values with float precision, we only consider a finite number $q \leq 256$ of different discrete brightness values. Since this introduces an error to the known data, it is a lossy preprocessing step to the lossless entropy coding. In a PDE-based setting, the benefits of grey value optimisation (GVO) can be diminished if such a quantisation is applied afterwards. Therefore, grey values should be optimised under the constraint of the coarse quantisation (see [23]).

To this end, we propose *quantisation-aware GVO* in Algorithm 1. Since both PDEs we consider are linear, we can use so-called inpainting echos [15] to speed up the optimisation. For a given mask $\boldsymbol{c}$ and corresponding grey values $\boldsymbol{g}$, let $r(\boldsymbol{c}, \boldsymbol{g})$ denote the inpainting result according to Section 2. Then, an echo is computed as $r(\boldsymbol{c}, \boldsymbol{e(i)})$ where in the image $\boldsymbol{e(i)}$ the $i$th known pixel is set to 1 and all other known data is set to zero. Thereby, each echo represents the influence of a single known data point. As long as the mask $\boldsymbol{c}$ is constant, we can compute the reconstruction for arbitrary grey values in $\boldsymbol{g}$ as $r(\boldsymbol{c}, \boldsymbol{g}) = \sum_{i \in K} g(\boldsymbol{x}) r(\boldsymbol{c}, \boldsymbol{e(i)})$. Note that the echoes only need to be computed once.

During optimisation, a Gauss-Seidel scheme successively updates the grey values at mask positions one by one. The crucial difference to [15] is that we directly quantise the grey values after every update. In our experiments, we observe that this algorithm already converges after only a few iterations over all mask points. Since the inpainting mask remains constant, the echoes can be reused for arbitrary quantisation parameters $q$. Thus, in contrast to the nonlinear inpainting case in R-EED [23], we are able to optimise $q$ thoroughly and efficiently with the help of a simple grid search.

The number $q$ of quantised grey values also influences the overall file size, since the entropy coding of the grey values becomes more efficient for smaller numbers of different grey values. Therefore, for smaller $q$, also the file size becomes smaller in general. Simultaneously, the error increases, since the constraints to the GVO become more strict. For a given mask, the best trade-off between file size and reconstruction quality must be found, i.e. the inpainting error and the file size have to be minimised simultaneously. For a given quantisation $q$, let $s : \{0, ..., 255\} \to \mathbb{N}$ be the file size in byte and $e : \{0, ..., 255\} \to \mathbb{R}$ the corresponding mean square error. By normalising both quantities to the range $[0, 1]$ and combining them additively, we define the *trade-off coefficient* $\mu$:

$$\mu := \frac{s(q)}{s(255)} + \frac{e(q)}{e(255)}.$$

The smaller this coefficient, the better the trade-off for a given $q$. Our goal is to find the best $q$ for a given mask. In our algorithms, we minimise $\mu$ with respect to $q$ in combination with quantisation-aware GVO.

**Efficient Representations of Positional Data.** In the previous paragraphs we have covered how compression steps influence tonal optimisation. In the following sections we propose two different approaches to perform the spatial optimisation: in Section 4, we allow free choice of point positions on the pixel grid and in Section 5 we restrict ourselves to a coarser, locally adaptive mesh. For both cases we discuss efficient storage. Note that for both codecs, decompression follows the same pattern: first we extract the entropy-coded mask data, then we apply inpainting to reconstruct the image.

## 4 Encoding with Exact Masks

Finding optimal positions on the pixel grid for a fixed amount of mask points is nontrivial. Besides the greedy approaches in [15], the algorithms in [2, 6] find masks by formulating an optimisation problem. As proposed in these papers, a primal-dual scheme can be employed for finding a solution. While [6] focuses on homogeneous diffusion, biharmonic inpainting is considered in [2]. Thus, we use both PDEs to find optimised masks for our compression scheme. Note that we use the variants of both methods that produce binary masks.

An inpainting mask acquired from the aforementioned algorithms is essentially a binary image of the same size as the original image. This suggests to use compression schemes like JBIG [10] to store the mask, in particular since it has been successfully applied in edge-based compression [16]. However, JBIG is optimised for binary images that obey a connectivity, since its primary intention was the use in fax machines. This is not the case for sparse inpainting masks which makes them hard to compress for JBIG.

Therefore, we have also evaluated other compression methods for binary image data. While block coding schemes [3,17,26,27] and coordinate coding [17] are not competitive to JBIG on their own, they can act as a preprocessing step for entropy coders such as Huffman Coding [8], arithmetic coding [21] and PAQ [13]. We experimentally found that a simple block coding scheme [27] in combination with PAQ is a good choice. It reduces the file size by up to 10% in comparison to JBIG. Together with the grey value and quantisation optimisation from Section 3, we obtain the following three-step compression for exact masks:

1. Select $p\%$ of total pixels with the optimal control approach [6].
2. Perform GVO with quantisation optimisation (Algorithm 1).
3. Optimise block size for optimal compression with PAQ.

## 5 Encoding with Stochastic Tree-Building

In this section, we pursue an approach that imposes restrictions to spatial optimisation. To define an adaptive regular grid, we start by storing a fixed point pattern for a given image: its four corner points and its midpoint. We can refine this grid by splitting the image in half in its largest dimension and add the same point pattern to its two subimages. Each split in this partitions the original image into smaller subimages and thereby refines the grid locally.

Compared to the free choice of the previous sections, this restriction reduces the size of the search space for optimal known data at the potential cost of reconstruction quality. In addition, it offers a lower coding cost for the locations by using *binary tree representations*. We represent each subimage of a given

```
Input: Original image $\boldsymbol{f}$, fraction $\alpha$ of tree nodes used as candidates for densification,
    fraction $\beta$ of candidate nodes that are added in each iteration, desired final mask
    density $d$.
Initialisation: Splitting tree $T$ containing only the root node $t_0$. Initial leaf node set
    $L := \{t_1, t_2\}$ containing child nodes of $t_0$.
Compute:
    Do
      1. Compute reconstruction $\boldsymbol{u}$ from mask $C(T)$ and image data $\boldsymbol{f}$.
      2. Choose randomly a candidate set $A \subset L$ containing $\alpha \cdot |L|$ nodes.
      3. For all $t_i \in A$ compute the subimage error $e(t_i)$.
      4. Add a subset of $\beta \cdot |A|$ candidate nodes $t_i$ with the largest errors $e(t_i)$ to the
         tree $T$.
      5. Update $L$ to contain all children of single split nodes from $T$.
    while $|C(T)| < d \cdot |\Omega|$.
Output: Tree $T$ with corresponding mask $C(T)$ of density $d$.
```

Algorithm 2: Stochastic tree densification.

```
Input: Original image $\boldsymbol{f}$, binary tree $T$, parameters $n < m$.
Compute: Repeat
      1. Create a backup copy $T_{\text{old}}$ of the splitting tree $T$.
      2. Compute reconstruction $\boldsymbol{u}_{\text{old}}$ from mask $C(T_{\text{old}})$ and image data $\boldsymbol{f}$.
      3. Remove the children of $n$ randomly chosen single split nodes from $T$.
      4. Randomly select a set $A$ containing $m$ leaf nodes from $T$.
      5. For all $t_i \in A$ compute the subimage error $e(t_i)$.
      6. Add the children of the $n$ nodes with the largest error $e(t_i)$ to $T$.
      7. Compute reconstruction $\boldsymbol{u}$ from mask $C(T)$ and image data $f$.
      8. If $\text{MSE}(\boldsymbol{u}, \boldsymbol{f}) > \text{MSE}(\boldsymbol{u}_{\text{old}}, \boldsymbol{f})$
             Reset changes, i.e. $T = T_{\text{old}}$.
    until number of maximum iterations is reached.
Output: Optimised tree $T$.
```

Algorithm 3: Nonlocal node exchange.

partition as a leaf node of the tree. For a tree $T$ consisting of nodes $t_0, \ldots, t_n$, the root node $t_0$ stands for the original image and the root's children for the halves of the original. By adding more nodes to the tree, the image corresponding to their parent node is split further. We can consider *leaf nodes* as indicators for termination, i.e.the subimage corresponding to a leaf node is not split further. This allows us to represent the tree as a bit sequence (0 for leaf nodes, 1 for other nodes). In [23], such an efficient representation of the point positions is obtained with a heuristic method. We propose to use a more powerful stochastic mask selection approach inspired by Mainberger et al. [15] instead.

The original idea in [15] for unrestricted point sets is to start with a full mask that contains all image points. From this mask, we remove a fixed percentage $\alpha$ of known data. After inpainting with the smaller mask, we add a fraction $\beta$ of the removed pixels with the highest reconstruction error back to the mask. This sparsification algorithm iterates the aforementioned steps until the target mask density is reached.

If we transfer this concept to a binary tree representation, there are some key differences: We have experimentally determined that densification is more efficient for tree structures than sparsification. Therefore, we start with a small amount of data and iteratively add more points at locations with large error until the target density is reached. In addition, we consider to add nodes to the tree instead of dealing with mask points directly. The tree structure dictates that only subimages corresponding to leaf nodes may be split. Such a split is equivalent to adding two child nodes to the former leaf node, each corresponding to one new subimage. Note that several mask points might be added by a single subdivision and that these mask points can also be contained in several of the neighbouring subimages. For our probabilistic method, we also need an appropriate error computation. In order to avoid a distortion of the influence of each node, we do not consider the mean square error in each subimage, but the sum $e(t_k)$ of unnormalised squared differences

$$e(t_k) = \sum_{(i,j)\in\Omega_k} (f_{i,j} - u_{i,j})^2 \qquad (2)$$

where $\Omega_k$ denotes the image domain of the subimage corresponding to the tree node $t_k$. Without this unnormalised error measure, the same per-pixel-error in small subimages would be weighted higher than in large subimages. Taking all of these differences into account, we define *stochastic tree densification* in Algorithm 2. For a target density $d$, it produces an optimised tree $T$ with a corresponding pixel mask $C(T) \subset \Omega$.

Just as the original sparsification algorithm, there is a risk that Algorithm 2 is caught in a local minimum. To avoid this problem, we propose an adapted version of the nonlocal pixel exchange from [15] in Algorithm 3. The method from [15] first generates a candidate set containing $m$ non-mask pixels. Afterwards, the $n$ pixels with the largest reconstruction error are exchanged with $n$ randomly chosen mask pixels. If the new mask yields a better reconstruction it is kept, otherwise the change is reverted.

In our case, we have to adapt the node selection to respect the tree structure in order to define a nonlocal node exchange. In particular, the candidate set that is removed from the tree can only consist of nodes that are only split once. This is the case if and only if both children are leaf nodes. We call these nodes *single split node* and the reversion of their associated split comes down to removing their children which converts the single split node to a leaf node. These modifications lead to Algorithm 3.

The binary trees obtained from the densification and nonlocal node exchange can finally be stored as a sequence of bits. As in [23], we store a maximum and minimum tree depth and only save the node-structure explicitly inbetween. As header data, only the image size and the number of quantised grey values have to be stored. In total, this leads to the following encoding procedure:

1. Select a fraction $p$ of total pixels with tree densification (Algorithm 2).

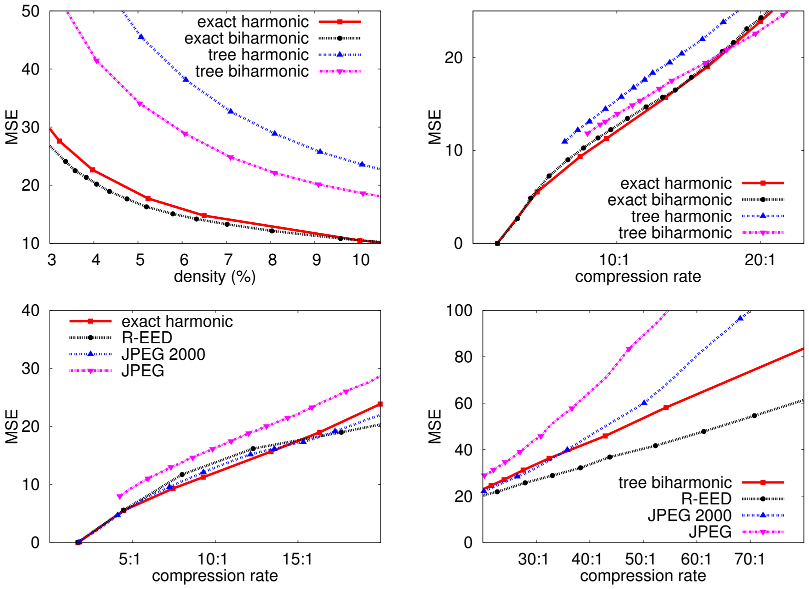

**Fig. 1.** Comparisons for the test image 256 × 256 image *peppers*. The top row compares harmonic and biharmonic versions of our codecs, the bottom row compares our best methods to transform coders and R-EED. **(a) Top Left:** Comparison at same mask density. **(b) Top Right:** Comparison at same compression ratio. **(c) Bottom Left:** Low to medium compression ratios. **(d) Bottom Right:** High compression ratios.

2. Optimise the splitting tree with nonlocal node exchange (Algorithm 3).
3. Perform GVO with quantisation optimisation (Algorithm 1).
4. Concatenate header, positional, and grey value data and apply PAQ.

## 6 Experiments

In the following we evaluate the capabilities of harmonic and biharmonic inpainting for the two compression methods from the previous sections. Our experiments rely on a set of widely-used test images. We start with a pure comparison of the inpainting operators on the test image *peppers*: We optimise exact and restricted masks with different densities, perform GVO and compare the mean square error (MSE) at the same mask density. The results in Fig. 1 (a) show that, in general, biharmonic performs better than harmonic inpainting given the same amount of known data. This is consistent with previous results [2]. The restriction of the mask to an adaptive grid has a significant negative impact on the quality. This affects harmonic inpainting more than its biharmonic counterpart.

Interestingly, an evaluation of the actual compression performance with the codecs from sections 4 and 5 in Fig. 1(b) shows a significantly different ranking than in the density comparison. For exact masks, harmonic inpainting can even

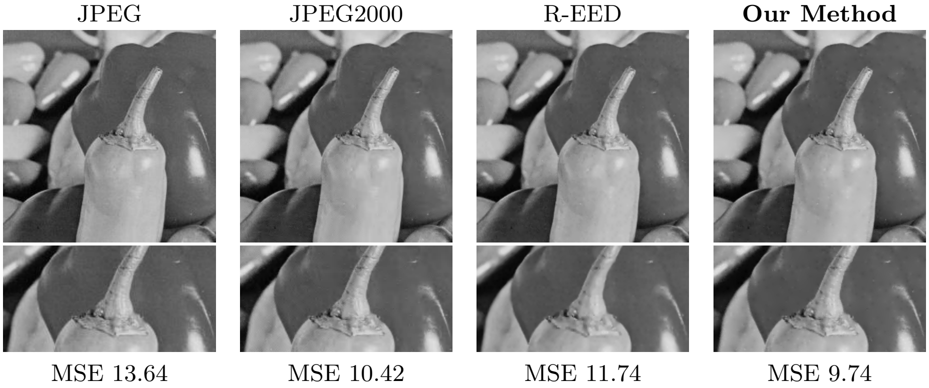

**Fig. 2.** Compression results for *peppers* (256 × 256 pixels) with compression ratio ≈ 8:1.

**Table 1.** MSE comparison on several test images. For the compression rate of 15:1 we use exact masks with homogeneous inpainting. The high compression rate of 60:1 is obtained with the biharmonic tree codec.

| Ratio | ≈ 15 : 1 | | | | ≈ 60 : 1 | | | |
|---|---|---|---|---|---|---|---|---|
| Image | elaine | lena | trui | walter | elaine | lena | trui | walter |
| JPEG | 34.69 | 20.06 | 16.24 | 6.73 | 77.39 | 97.37 | 116.98 | 69.40 |
| JPEG2000 | 31.23 | 14.73 | 12.27 | 5.70 | 54.18 | 60.48 | 88.96 | 47.09 |
| R-EED | 35.48 | 16.56 | 11.27 | 5.48 | 49.87 | 56.96 | 44.38 | 24.05 |
| Our Method | 31.38 | 17.00 | 14.85 | 6.94 | 55.96 | 62.01 | 60.92 | 34.95 |

surpass its biharmonic counterpart. The coding cost for the known data is similar in both cases, but since harmonic inpainting is less sensitive to a coarse quantisation of the grey values, it performs overall better than biharmonic inpainting. The drawbacks of the restrictions in the tree-based approach are attenuated by the reduced positional coding cost. After a break-even point around ratio 20:1, the biharmonic tree-based method outperforms both exact approaches.

In relation to transform-based coders, the tree-based method performs consistently better than JPEG and in many cases also outperforms JPEG2000 for compression rates larger than 35:1. Nevertheless, R-EED copes better with the restriction to the adaptive grid and remains the preferable choice at high compression ratios. In comparison to other PDE-based methods, linear diffusion performs best in the area of low to medium compression rates (up to 15:1). Fig. 1 and Table 1 show that, it can beat R-EED, mostly outperforms JPEG, and comes close to the quality of JPEG2000. Only on images that contain partially smooth data with high-contrast edges, R-EED is significantly better (e.g. on *trui* and *walter*). This demonstrates how powerful simple PDEs can be.

## 7 Conclusion

We have shown that codecs with parameter-free linear inpainting PDEs can beat both the quasi standard JPEG2000 of transform-based compression and the state-of-the-art in PDE-based compression. Comparing the different inpainting operators yields a valuable general insight: *The performance of PDEs for compression can only be evaluated in the context of actual codecs.* Comparisons that do not consider all compression steps can lead to false rankings of inpainting operators that do not reflect their real compression capabilities. In particular, the sensitivity of the biharmonic operator to coarsely quantised known data makes the simpler harmonic diffusion the preferable choice for compression.

Currently, the algorithms for mask selections are not competitive to JPEG2000 in regard to runtime. In our ongoing research, we focus on speeding up our algorithms for mask selection, such that PDE-based codecs are not only fast for decompression [11], but also for compression. Moreover, we will consider combinations of linear PDEs with patch-based inpainting for highly textured images, since this concept is already successful for nonlinear PDEs [20].

## References

1. Belhachmi, Z., Bucur, D., Burgeth, B., Weickert, J.: How to choose interpolation data in images. SIAM Journal on Applied Mathematics 70(1), 333–352 (2009)
2. Chen, Y., Ranftl, R., Pock, T.: A bi-level view of inpainting-based image compression. In: Proc. 19th Computer Vision Winter Workshop. pp. 19–26. Křtiny, Czech Republic (Feb 2014)
3. Fränti, P., Nevalainen, O.: Compression of binary images by composite methods based on block coding. Journal of Visual Communication and Image Representation 6(4), 366–377 (1995)
4. 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, Lecture Notes in Computer Science, vol. 3752, pp. 37–48. Springer, Berlin (2005)
5. Gautier, J., Meur, O.L., Guillemot, C.: Efficient depth map compression based on lossless edge coding and diffusion. In: Proc. 29th Picture Coding Symposium. pp. 81–84. Krakow, Poland (May 2012)
6. Hoeltgen, L., Setzer, S., Weickert, J.: An optimal control approach to find sparse data for Laplace interpolation. In: Heyden, A., Kahl, F., Olsson, C., Oskarsson, M., Tai, X.C. (eds.) Energy Minimization Methods in Computer Vision and Pattern Recognition, Lecture Notes in Computer Science, vol. 8081, pp. 151–164. Springer, Berlin (2013)
7. Hoffmann, S., Mainberger, M., Weickert, J., Puhl, M.: Compression of depth maps with segment-based homogeneous diffusion. In: Kuijper, A., Bredies, K., Pock, T., Bischof, H. (eds.) Scale-Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, vol. 7893, pp. 319–330. Springer, Berlin (2013)
8. Huffman, D.A.: A method for the construction of minimum redundancy codes. Proceedings of the IRE 40, 1098–1101 (1952)

9. Iijima, T.: Basic theory on normalization of pattern (in case of typical one-dimensional pattern). Bulletin of the Electrotechnical Laboratory 26, 368–388 (1962), in Japanese
10. Joint Bi-level Image Experts Group: Information technology progressive lossy/lossless coding of bi-level images. Standard, International Organization for Standardization (1993)
11. Köstler, H., Stürmer, M., Freundl, C., Rüde, U.: PDE based video compression in real time. Tech. Rep. 07-11, Lehrstuhl für Informatik 10, Univ. Erlangen–Nürnberg, Germany (2007)
12. Li, Y., Sjostrom, M., Jennehag, U., Olsson, R.: A scalable coding approach for high quality depth image compression. In: 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video. pp. 1–4. Zurich, Switzerland (Oct 2012)
13. Mahoney, M.: Adaptive weighing of context models for lossless data compression. Tech. Rep. CS-2005-16, Florida Institute of Technology, Melbourne, Florida (Dec 2005)
14. Mainberger, M., Bruhn, A., Weickert, J., Forchhammer, S.: Edge-based compression of cartoon-like images with homogeneous diffusion. Pattern Recognition 44(9), 1859–1873 (2011)
15. Mainberger, M., Hoffmann, S., Weickert, J., Tang, C.H., Johannsen, D., Neumann, F., Doerr, B.: Optimising spatial and tonal data for homogeneous diffusion inpainting. In: Bruckstein, A., ter Haar Romeny, B., Bronstein, A., Bronstein, M. (eds.) Proc. Third International Conference on Scale Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, vol. 6667, pp. 26–37. Springer, Berlin (Jun 2011)
16. Mainberger, M., Weickert, J.: Edge-based image compression with homogeneous diffusion. In: Jiang, X., Petkov, N. (eds.) Computer Analysis of Images and Patterns, Lecture Notes in Computer Science, vol. 5702, pp. 476–483. Springer, Berlin (2009)
17. Mohamed, S.A., Fahmy, M.: Binary image compression using efficient partitioning into rectangular regions. IEEE Transactions on Communications 43(5), 1888–1893 (1995)
18. Pennebaker, W.B., Mitchell, J.L.: JPEG: Still Image Data Compression Standard. Springer, New York (1992)
19. Peter, P., Weickert, J.: Colour image compression with anisotropic diffusion. In: Proc. 21st IEEE International Conference on Image Processing. pp. 4822–4862. Paris, France (Oct 2014)
20. Peter, P., Weickert, J.: Compressing images with diffusion- and exemplar-based inpainting. In: Aujol, J.F., Nikolova, M., Papadakis, N. (eds.) Scale-Space and Variational Methods in Computer Vision, Lecture Notes in Computer Science, vol. 9087, pp. 154–165. Springer, Berlin (2015)
21. Rissanen, J.J.: Generalized Kraft inequality and arithmetic coding. IBM Journal of Research and Development 20(3), 198–203 (1976)
22. Schmaltz, C., Mach, N., Mainberger, M., Weickert, J.: Progressive modes in PDE-based image compression. In: Proc. 30th Picture Coding Symposium. pp. 233–236. San Jose, CA (Dec 2013)
23. Schmaltz, C., Peter, P., Mainberger, M., Ebel, F., Weickert, J., Bruhn, A.: Understanding, optimising, and extending data compression with anisotropic diffusion. International Journal of Computer Vision 108(3), 222–240 (Jul 2014)
24. Taubman, D.S., Marcellin, M.W. (eds.): JPEG 2000: Image Compression Fundamentals, Standards and Practice. Kluwer, Boston (2002)

25. Weickert, J.: Theoretical foundations of anisotropic diffusion in image processing. Computing Supplement 11, 221–236 (1996)
26. Zahir, S., Naqvi, M.: A near minimum sparse pattern coding based scheme for binary image compression. In: Proc. 2005 IEEE International Conference on Image Processing. vol. 2, pp. II–289. Genova, Italy (Sep 2005)
27. Zeng, G., Ahmed, N.: A block coding technique for encoding sparse binary patterns. IEEE Transactions on Acoustics, Speech, and Signal Processing 37(5), 778–780 (1989)