Saarland University
Faculty of Mathematics and Computer Science
Mathematical Image Analysis Group (MIA)

Competitive Image Compression with Linear PDEs

A Dissertation Submitted Towards the Degree Doctor of Engineering (Dr.-Ing.)
of the Faculties of Mathematics and Computer Science of Saarland University

submitted by

Sebastian Hoffmann

Saarbrücken

December, 2016

Day of Colloquium
June 16, 2017

Dean of Faculty
Prof. Dr. Frank-Olaf Schreyer

Chair of the Committee
Prof. Dr. Philipp Slusallek

Reviewers
Prof. Dr. Joachim Weickert
Prof. Dr. Gerlind Plonka-Hoch

Academic Assistant
Dr. Pascal Peter

Short Abstract

In recent years, image compression methods with partial differential equations (PDEs) have become a promising alternative to standard methods like JPEG or JPEG2000. They exploit the ability of PDEs to recover an image from only a small amount of stored pixels. To do so, this information is diffused into unknown image regions. Most successful codecs rely on a sophisticated nonlinear diffusion process. In this thesis, we investigate the potential of linear PDEs for image compression. This follows up recent promising compression results with the Laplace and biharmonic operator. As a central contribution, we thoroughly study the diffusion-based image reconstruction, and establish a connection to the concept of sparsity with the help of discrete Green's functions. Apart from gaining novel theoretical insights, this offers algorithmic benefits: We obtain a fast reconstruction procedure and improved data optimisation methods. The power of linear PDEs is demonstrated by proposing three different compression approaches. Besides a designated codec for depth maps we present an algorithm for general images. The third approach aims at a fast decoding. Experiments show that linear PDEs can compete with nonlinear PDEs in a compression context, and that codecs with linear PDEs are even able to outperform standard approaches in terms of both speed and quality.

Kurzzusammenfassung

In den letzten Jahren sind Bildkompressionsmethoden basierend auf partiellen Differentialgleichungen (PDEs) eine vielversprechende Alternative zu Standardmethoden wie JPEG oder JPEG2000 geworden. Diese nutzen die Fähigkeit von PDEs aus, ein Bild von nur einer kleinen Menge an gespeicherten Pixeln rückgewinnen zu können. Die meisten erfolgreichen Codecs verwenden einen aufwändigen nichtlinearen Diffusionsprozess. In dieser Arbeit werden wir das Potential von linearen PDEs zur Bildkompression untersuchen. Dies schließt an die kürzlichen vielversprechenden Kompressionsergebnisse mit dem Laplace und biharmonischen Operator an. Als zentralen Beitrag studieren wir die PDE-basierte Bildrekonstruktion und stellen mit Hilfe von diskreten Greenschen Funktionen eine Verbindung zum Sparsity-Konzept her. Dies bietet neben neuen theoretischen Erkenntnissen auch algorithmische Vorteile: Wir erhalten ein schnelles Rekonstruktionsverfahren und verbesserte Datenoptimierungsmethoden. Die Stärke von linearen PDEs wird gezeigt, indem wir drei verschiedene Kompressionsmethoden vorschlagen. Neben eines designierten Codecs für Tiefenkarten präsentieren wir einen Algorithmus für allgemeine Bilder. Der dritte Ansatz zielt auf eine schnelle Dekodierung ab. Experimente zeigen, dass lineare PDEs mit nichtlinearen PDEs in einem Kompressionskontext konkurrieren können und dass Codecs mit linearen PDEs in Bezug auf Geschwindigkeit und Qualität sogar Standardmethoden übertreffen können.

Abstract

Image compression plays an important role in our nowadays lives. For instance, the enormous growth of image data on the internet necessitates advanced codecs which allow for an efficient storage and transmission. Besides the commonly used standard JPEG codec [150] and its successor JPEG2000 [191], image compression methods relying on partial differential equations (PDEs) have gained more and more attention in recent years. Instead of storing the coefficients of the discrete cosine or wavelet transform, the strategy of PDE-based methods is to encode the sparse image information of a few cleverly selected pixels. The power of the PDEs lies in recovering the image in the decoding phase. In this image inpainting step, the known information is propagated into the missing regions with the help of a diffusion process. Clearly, the reconstruction does not only depend on the selected pixels, but also on the employed diffusion process.

In the context of PDE-based compression, the current state-of-the-art method [170] relies on inpainting with the nonlinear edge-enhancing anisotropic diffusion (EED) [204]. Despite its remarkable reconstruction capability, this sophisticated process suffers from several issues. First of all, the incorporated parameters need to be tuned to obtain good results. Secondly, the inpainting process involves successive updates of the nonlinearities and is thus rather time-consuming. Finally, the nonlinearity leads to a nontrivial dependency of the prescribed data on the reconstruction. This tremendously complicates the task of selecting optimal pixel data to be stored. Along with the slow inpainting, a cumbersome optimisation procedure is required. Despite their qualitatively good compression performance, PDE-based compression methods are hence far from being as fast as standard codecs.

In this thesis, we evaluate the potential of linear PDEs for image compression. In contrast to nonlinear ones, they benefit from a much faster computation and are parameter-free. Moreover, we will see that the linearity tremendously facilitates data optimisation tasks. Even though these conceptually simpler diffusion processes usually do not achieve the same reconstruction quality as EED, promising compression results have been achieved in the past with the Laplacian or biharmonic operator [125, 80, 118], at least for specific types of images. By going a step further, we investigate if linear PDEs can even compete with EED for the compression of general images, and if they are able to keep up with standard approaches in terms of both quality and runtime aspects.

A central part is the thorough investigation of the inpainting process itself. As a result, we find that there is a close connection to the concept of sparsity. In particular, discrete Green's functions turn out to represent atoms in a dictionary which can be used to describe the inpainting solution. This does not only offer novel theoretical insights, but also yields algorithmic benefits. For instance, the discrete Green's functions allow for a very fast image reconstruction process. To this end, we elaborate efficient ways to construct them and to perform dedicated operations.

The data optimisation plays a crucial role for the performance of the compression algorithm. Hence, it is indispensable to explore and compare potential ways to perform this task. It does not only make sense to find the optimal locations of the prescribed pixels, but also the corresponding tonal values. This enables to obtain the globally best reconstruction by tolerating larger local errors. Discrete Green's functions allow for a very fast tonal optimisation for smaller amounts of prescribed pixels, while methods relying on the standard inpainting formulation perform better for higher pixel densities. From the latter approaches, we find that a proposed gradient descent scheme can outperform the primal-dual approach in [93]. Inpainting echoes are useful for incorporating a quantisation into the optimisation, and also for extending the tonal optimisation for EED inpainting. Considering the spatial optimisation, the best method usually depends on the specifications. The qualitative best results are obtained with the optimal control framework [92]. However, this requires to determine a parameter to obtain the desired number of pixels. Probabilistic approaches turn out to be a good alternative which can even be applied for arbitrary inpainting processes. Good results for inpainting with the Laplacian are also obtained by the theory of Belhachmi *et al.* [23] which has real-time capabilities. Beyond this, discrete Green's functions even enable a continuous optimisation of the locations.

The potential of linear PDEs for image compression is finally demonstrated by presenting three different codecs. First, a designated codec for depth maps is proposed. By incorporating segment boundaries into the inpainting process, remarkable compression results can be achieved. Experiments show that this method is superior to JPEG and JPEG2000, and can even compete with the designated future standard HEVC. The second codec targets on the compression of general images. With the help of highly optimised data, one can show that linear PDEs can yield a comparable compression performance as EED for medium to low compression ratios. This approach is even able to outperform JPEG and JPEG2000 on general images. While focusing on the quality in the first two codecs, the third one aims at being competitive with JPEG in terms of both quality and speed. To this end, we adopt some techniques from JPEG such as a block decomposition of the image domain. In the end, experiments show that linear PDEs enable to compete with JPEG on general colour images in both criteria.

Zusammenfassung

Bildkompression spielt eine wichtige Rolle in unserem alltäglichen Leben. Das enorme Wachstum an Bilddaten, beispielsweise im Internet, erfordert fortgeschrittene Codecs, die ein effizientes Speichern und Übertragen erlauben. Neben dem gemeinhin genutzen Standard JPEG Codec [150] und dessen Nachfolger JPEG2000 [191] haben in den letzten Jahren Bildkompressionsverfahren, die auf partiellen Differentialgleichungen (PDEs) beruhen, immer mehr Beachtung erlangt. Anstatt die Koeffizienten der diskreten Kosinus- oder Wavelet-Transformation zu speichern, ist die Strategie der PDE-basierten Verfahren, die Bildinformation von einigen wenigen klug gewählten Pixeln zu kodieren. Die Stärke der PDEs liegt in der Wiedergewinnung des Bildes in der Dekodierungsphase. In diesem Bildinterpolationsschritt wird die bekannte Information mit Hilfe eines Diffusionsprozesses in die unbekannten Bereiche zerstreut. Es ist klar, dass die Rekonstruktion nicht nur von den gewählten Pixeln abhängt, sondern auch vom verwendeten Diffusionsprozess.

Im Zusammenhang von PDE-basierter Kompression beruht der aktuelle Stand der Technik [170] auf Interpolation mit der nichtlinearen, kantenverstärkenden, anisotropen Diffusion (EED) [204]. Trotz seines bemerkenswerten Rekonstruktionsvermögens leidet dieser komplexe Prozess unter einigen Nachteilen. Zunächst bedarf es einer Optimierung der beinhalteten Parameter, um gute Ergebnisse zu erzielen. Zweitens bringt der Interpolationsprozess fortlaufende Aktualisierungen der Nichtlinearitäten mit sich und ist daher eher zeitaufändig. Schlussendlich führen die Nichtlinearitäten zu einem nichttrivialen Zusammenhang zwischen den vorgeschriebenen Daten und der Rekonstruktion. Das erschwert es ungemein, die zur Speicherung vorgesehenen optimalen Pixeldaten auszuwählen. Zusammen mit der langsamen Interpolation bedingt dies einen aufwändigen Optimierungsprozess. Trotz ihrer qualitativ guten Kompressionsleistung sind diese PDE-basierte Verfahren weit entfernt von der Schnelligkeit von Standardcodecs.

In dieser Arbeit evaluieren wir das Potential von linearen PDEs zur Bildkompression. Im Gegensatz zu den Nichtlinearen profitieren diese von einer deutlich schnelleren Berechnung und sind frei von Parametern. Außerdem werden wir sehen, dass die Linearität die Datenoptimierungsaufgaben deutlich erleichtert. Obwohl diese konzeptuell einfachen Diffusionsprozesse gewöhnlich nicht die gleiche Rekonstruktionsqualität wie EED erreichen, wurden in der Vergangenheit vielversprechende Kompressionsergebnisse mit dem Laplace oder biharmonischen Operator zumindest für spezielle Bildtypen erreicht [125, 80, 118]. Davon ausgehend untersuchen wir, ob lineare PDEs sogar bei der Kompression von allgmeinen Bildern mit EED konkurrieren können und ob sie fähig sind, in Bezug auf Qualitäts- sowie Laufzeitaspekten mit Standardansätzen mitzuhalten.

Ein zentraler Teil ist die umfassende Untersuchung des Interpolationsprozesses selbst. Als ein Ergebnis finden wir heraus, dass es eine enge Verbindung zu dem

Sparsity-Konzept gibt. Insbesondere stellen sich diskrete Greensche Funktionen als Teile eines Lexikons heraus, mit deren Hilfe die Interpolationslösung beschrieben werden kann. Dies bietet nicht nur neue theoretische Einsichten, sondern bringt auch algorithmische Vorteile mit sich. Zum Beispiel erlauben die diskreten Greenschen Funktionen einen sehr schnellen Bildrekonstruktionsprozess. Dazu erarbeiten wir effiziente Wege, um sie zu konstruieren und dazugehörige Operationen durchzuführen.

Die Datenoptimierung spielt eine entscheidende Rolle für die Performanz des Kompressionsalgorithmus. Daher ist es unabdingbar, potentielle Ansätze für diese Aufgabe zu erkunden und zu vergleichen. Es macht nicht nur Sinn, die optimalen Stellen der Pixel zu optimieren, sondern auch die entsprechenden tonalen Werte. Dies erlaubt es, die global beste Rekonstruktion zu erhalten, indem größere lokale Fehler toleriert werden. Die diskreten Greenschen Funktionen erlauben eine sehr schnelle tonale Optimierung bei kleineren Mengen vorgegebenen Pixeln, während Methoden basierend auf der Standard-Interpolationsformulierung bei höheren Pixeldichten besser sind. Bei den letzteren Ansätzen finden wir heraus, dass ein Gradientenabstiegsschema das Primal-Dual Verfahren [93] schlagen kann. Interpolationsechos sind nützlich, um eine Quantisierung in die Optimierung einzubeziehen, also auch um die tonale Optimierung für EED zu erweitern. Was die örtliche Optimierung betrifft hängt die beste Methode gewöhnlich von den Anforderungen ab. Die qualitativ besten Ergebnisse werden mit dem Ansatz der optimalen Kontrolle [92] erzielt. Dieser erfordert jedoch eine Parameterwahl um die gewünschte Anzahl an Pixel zu erhalten. Probabilistische Ansätze erweisen sich als gute Alternative, die sogar für beliebige Interpolationsverfahren angewendet werden können. Gute Ergebnisse für den Laplace erhält man auch mit der Theorie von Belhachmi *et al.* [23], welche Echtzeitpotenzial hat. Diskrete Greensche Funktionen erlauben darüber hinaus sogar eine kontinuierliche Optimierung der Pixelorte.

Das Potential von linearen PDEs für Bildkompression wird schlussendlich nachgewiesen, indem drei verschiedene Codecs präsentiert werden. Zuerst schlagen wir einen designierten Codec für Tiefenkarten vor. Durch das Einbinden von Segmentgrenzen in den Interpolationsprozess können erstaunliche Ergebnisse erzielt werden. Experimente belegen, dass diese Methode JPEG und JPEG200 überlegen ist und sogar mit dem vorgesehenen zukünftigen Standard HEVC mithalten kann. Der zweite Codec zielt auf die Kompression allgemeiner Bilder ab. Mit Hilfe von hochoptimierten Daten kann man zeigen, dass lineare PDEs für mittlere bis geringe Kompressionsverhältnisse eine vergleichbare Kompressionsleistung wie EED erzielen können. Dieser Ansatz kann sogar JPEG und JPEG2000 auf allgemeinen Bildern übertreffen. Während sich die ersten beiden Codecs auf die Qualität konzentrieren, zielt der dritte darauf ab, in Bezug auf Qualität und Laufzeit wettbewerbsfähig mit JPEG zu sein. Dazu übernehmen wir einige Techniken von JPEG, wie zum Beispiel die Blockunterteilung des Bildbereichs. Am Ende zeigen Experimente, dass lineare PDEs es ermöglichen, mit JPEG auf allgemeinen Farbbildern in beiden Kriterien mitzuhalten.

Acknowledgements

First of all, I want to thank my supervisor Prof. Joachim Weickert who has given me the opportunity to do my Ph.D. in the Mathematical Image Analysis (MIA) group. I am grateful for him providing me guidance and feedback now and then while at the same time allowing me enough freedom for my research. Moreover, I appreciate the generous working environment as well as the possibility to conduct autonomous teaching.

Furthermore, I want to thank all people who collaborated with me on various topics: First of all, Dr. Markus Mainberger gave me a lot of valuable advice in the initial phase. As roommates, we had many inspiring discussions which served as basis for several publications. Moreover, I want to thank Prof. Gerlind Plonka-Hoch from the university of Göttingen for the very pleasant collaboration on the topic of discrete Green's functions. I highly appreciate the continuous and constructive exchange of ideas. Besides the collaboration with Dr. Pascal Peter on PDE-based image compression, I also want to thank him for all the fruitful discussions that we had on various topics. With his Bachelor's thesis, Michael Puhl provided valuable input on the topic of segment-based diffusion inpainting. Moreover, I am pleased to have worked with Markus Schneider on his Master's thesis and the subsequent publication. I also had many interesting discussions with Dr. Laurent Hoeltgen.

Besides those direct collaborations, a big thank you goes to all other current and former colleagues at the MIA chair: Dr. Matthias Augustin, Naoufal Amrani, Sarah Andris, Leif Bergerhoff, Prof. Michael Breuß, Prof. Andrés Bruhn, Marcelo Cárdenas, Dr. Oliver Demetz, Silvano Galliani, Dr. Sven Grewenig, Dr. Pascal Gwosdek, David Hafner, Kai Hagenburg, Sabine Müller, Dr. Peter Ochs, Dr. Nico Persch, Dr. Christopher Schroers, Dr. Simon Setzer, Martin Schmidt and Dr. Christian Schmaltz. They all contributed to a very enjoyable and motivating working atmosphere.

Furthermore, I would like to express my graditude to the secretary Ellen Wintringer. She was always there to rescue me from any organisational or bureaucratic problem I had. Also, I thank the system administrators Marcus Hargarter, Heinz-Ulrich Griehl and Peter Franke for their help with all kinds of technical issues over the years.

Most importantly, my biggest thank goes to my family, in particular to Andrea and Bernd Hoffmann as well as to Lea Reiland. They always support me and are there for me when I need them. Without them I would not have been able to get this far.

Lastly, I gratefuly acknowledge the partial funding by the Deutsche Forschungsgemeinschaft (DFG) through a Gottfried Wilhelm Leibniz Prize for Prof. Joachim Weickert.

Contents

Chapter 1

Introduction

1.1 Image Compression

1.1.1 Motivation

In our nowadays times, modern technologies are becoming more and more popular. It is obvious that this trend has already long since entered our everyday private lives. People employ mobile phones, tablets, computers and other electronic devices to communicate over the internet: they send e-mails, exchange images and videos over social networks, do phone and video calls, watch movies on video on demand services or browse the internet, just to mention a few examples. Besides the private sector, modern software and hardware is also omnipresent in professional areas to facilitate work. For instance, physicians acquire CT or MRT scans of their patients for an improved treatment. In the future, it might even be possible to obtain a medical diagnosis online by sending photographs to a doctor over the internet. This list can steadily be continued.

It is clear that these developments go hand in hand with an enormous growth of digital image data which immediately leads to the following issues. First of all, one needs to be able to store this data on physical hardware. Although large memory capacities are available on modern computers, the limits are reached very soon when one does not pay attention to an efficient memory usage. Secondly, one should be able to exchange images over the internet in a reasonable time. Unfortunately, the limited bandwidth only allows for a certain amount of data to be transported within a given amount of time.

As an example, for a typical high definition (HD) colour image of size 1920×1200, one needs around 6.59 Megabytes to store it in a naïve way. This sounds like a small number, but this can quickly become a huge amount of data if one has to deal with more than only one image. For instance, 1000 images already require around 6.44 Gigabytes. On a holiday trip, this amount of images can quickly be acquired with a camera.

The mentioned issues can be approached by two conceptually different ways. One possibility is to improve the necessary hardware. This can include an increase of the memory capacity, or an extension of the available bandwidth. However, this strategy quickly reaches its limits. Another way to deal with the huge amount of data is to reduce the space it is occupying with a software solution. To this end, one can design specific algorithms to compress the data. Thereby, it is important that the essential information can be recovered from the data. Naturally, a joint improvement on a hardware and a software level is indispensable in the end.

In this thesis, we investigate methods for the compression of images on a software level. Before going into more detail, we start by discussing general image compression methods in the following, and consider existing approaches. Afterwards, we introduce the alternative concept of image compression with partial differential equations. Besides providing an overview of existing methods, the objectives and an overview of this thesis will be discussed.

1.1.2 Encoding and Decoding

Typical image compression methods consist of two main parts. The first part, called the *encoder*, converts the original image into a compact file representation. This file should have a considerably smaller size compared to the size of the original image file, and is thus perfectly suited for transmitting it over the internet or for storing it on a hard disk for example. The second part of a compression method is given by a so-called *decoder*. Its task is to retrieve an image only from the information provided by the small encoded file. In general, one distinguishes two different cases. If the compression method is *lossless*, the recovered image should be identical to the original image. Otherwise, if the compression is *lossy*, the reconstructed image is allowed to exhibit some degradations. In this case, the reconstruction should be as close as possible to the original image. This work mainly focuses on lossy image compression. An image compression algorithm is usually also denoted as *codec*, an abbreviation obtained by appending the first letters of the two words *co*der and *dec*oder.

It is clear that the encoding and decoding go hand in hand. The encoder is required to extract and store data in such a way that the decoder is able to process this data and recover a desired reconstruction. Due to this dependency, it makes sense to design them in a joint way to obtain the best possible performance. An important aspect is that the encoder is free to choose any data it wants as long as the decoder can reconstruct an image of desired quality. In particular, it does not have to stick to the data provided by the original image. In contrast, the data can be optimised in the encoding part in order to improve the compression performance.

1.1.3 Compression Performance

Being able to assess the performance of an image compression algorithm is often desired, especially when one wants to compare two different compression methods. In this context, the *compression ratio* is a standard measure that describes the amount of file size reduction that can be achieved by a codec. It is computed as the ratio between the file size of the original uncompressed image and the encoded smaller file. Hence, a larger compression ratio means that the file size reduction by the encoder is larger.

With the help of the compression ratio one can effectively compare the compression performance of two codecs: for the same reconstruction quality, the one with the higher compression ratio has the better compression efficiency. Obviously, this assessment involves the ability to judge the quality of the reconstructed image. While this is not an issue for lossless codecs where the images are supposed to be identical to the original image, one needs to have a meaningful quality measure in the case of a lossy compression.

One possibility to evaluate the reconstruction quality is to make a qualitative judgement based on the human visual perception. Although this visual inspection is the decisive criterion in most practical scenarios, this is very cumbersome to be performed in reality. As a remedy, one often employs a mathematical error measure which is computed for a given reconstruction. This quantitative measure tries to reflect the quality of the visual perception by ranking it on a certain scale. Some of these error measures can either be motivated from mathematics, others from the human visual perception.

1.1.4 Runtime Aspects

The primary aim of image compression algorithms is to have a high compression performance. However, an important aspect which should not be neglected is that they should also have a good runtime performance. The best image compression algorithm is useless if it takes forever to compress or decompress the data. Thereby, it is much more important to have a fast decoding than an efficient encoding in many applications. For example, the encoding of a movie typically lasts a very long time. This is not a big issue as this only has to be done once in the beginning before bringing the compressed file to a DVD. More importantly, it is crucial to have a fast decoding for being able to watch the movie in real-time.

Instead of using more advanced methods and algorithms to perform the compression tasks, one can also rely on more sophisticated hardware to improve the runtime. For instance, instead of performing computations on the *central processing unit* (CPU) which is naturally available on a computer, some methods employ a so called *graphics processing unit* (GPU). Besides the original purpose to process the output for the display, the GPU can also be used for other purposes. Due to the fact that a GPU contains many small processing units, it is possible to perform most computations in parallel. As such a GPU is also very common on modern computers, it is reasonable to employ this possibility, too. Nevertheless, as a GPU is not always available, our primary goal is to design methods which are suited for the CPU.

1.1.5 Standard Methods

In the field of image compression, there already exist some standard methods. The most frequently used compression method - in particular for lossy image compression - is the JPEG standard [150]. Its name is derived from its creator, namely the *Joint Photographic Experts Group*. The core idea of JPEG is to decompose the image into small blocks of size 8×8. Within these blocks, a discrete cosine transform (DCT) is applied. Then, the individual coefficients of the DCT basis are stored with different degrees of precision. Thereby, a higher precision is used for the low frequent parts, and vice versa. This mimics the human visual perception as high frequent components are less crucial for having a good impression of the image. Instead of the usual RGB colour space, the YCbCr colour space is employed for colour images. Due to the fact that the a human can distinctly see more detail in the brightness channel Y than in the hue and saturation channels Cb and Cr, the latter two channels are downsampled. JPEG supports both lossy and lossless compression.

There also exists a successor of JPEG, namely JPEG2000 [191]. Its compression performance clearly surpasses the one of JPEG and incorporates many more features such as a higher dynamic range support, a progressive mode or a more flexible file format adaptation. However, it has not been established as standard compression routine so far. The main reasons for this are the higher demand on computational power, patent issues as well as the missing support of the new format on many platforms. In contrast to JPEG, JPEG2000 relies on biorthogonal wavelets instead of the DCT. Thereby, it it does not divide the image into small blocks, but decomposes the whole image into its individual frequency bands. This avoids unpleasant artefacts at block boundaries.

Although JPEG and JPEG2000 are already highly sophisticated and perform very well in most scenarios, this is not the end of the road. Besides a potential improvement of their coding efficiency, they additionally reveal certain visual issues in the decoded images. For example, images compressed with JPEG or JPEG2000 often show so-called ringing artefacts which are visible especially around image edges. The reason for this is that both are transform-based methods which rely on basis functions. An error in the coefficients can thus lead to such unpleasant effects. Besides, colour shifts can occur, leading to undesired new colours. Moreover, as already indicated, JPEG is prone to suffer from blocking artefacts due to the individual treatment of the 8×8 blocks. An example of the aforementioned artefacts is depicted in Figure 1.1. Thus, there is an obvious need for improved image compression methods.

The High Efficiency Video Coding (HEVC), also known as H.265, is a designated future standard for video compression [189]. Even though this codec is mainly developed for the purpose of video coding, it also provides an intra-coding mode for the efficient compression of still images [145]. Similar to JPEG, it relies on the DCT and the YCbCr colour space, but uses more advanced coding techniques which base on a subdivision of the image into so-called coding tree units and coding tree blocks.

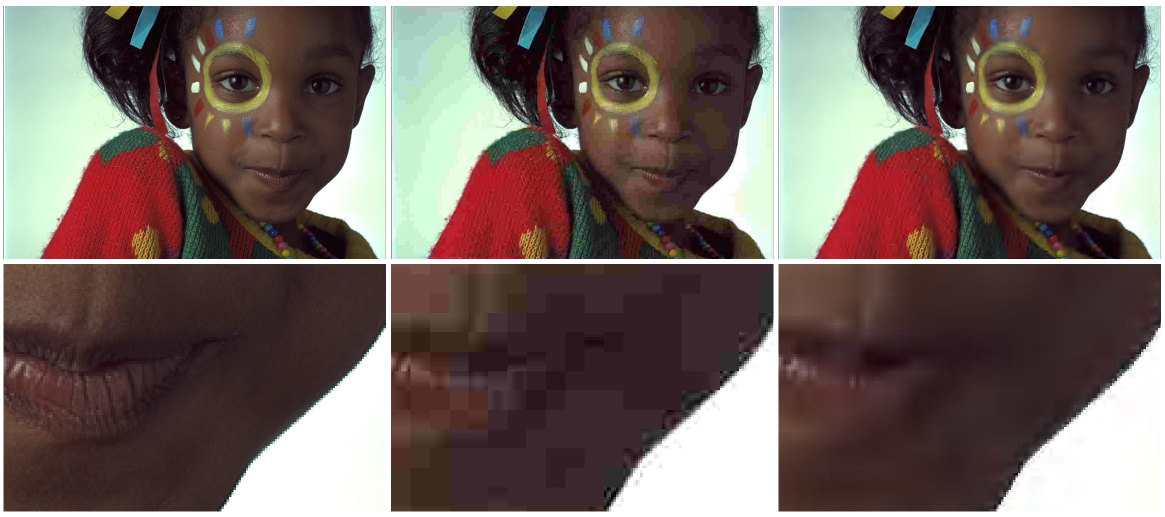

Figure 1.1. Top left: Original image (size: 768 × 512, source: Kodak image database [56]). **Top centre and right:** Compressed JPEG and JPEG 2000 image, respectively, with a compression ratio of 60 : 1. Both methods show visible artefacts. **Bottom row:** Zoom into images.

A promising alternative to these transform-based methods are image compression approaches which base on partial differential equations. They are discussed thoroughly in the following section.

1.2 PDE-based Image Compression

Image compression methods which base on partial differential equations (PDEs) are becoming increasingly successful. Thus, it is not surprising that they gain more and more popularity. The underlying core ingredient of PDE-based image compression methods is the so called image inpainting which is a powerful tool to interpolate sparse prescribed image data.

In the following the general concept of PDE-based inpainting methods is explained. Moreover, related image interpolation methods as well as methods which aim at recovering an image from sparse data are presented. Afterwards, it is shown how the PDE-based interpolation methods can be employed in the context of image compression. Existing methods in this area are presented, too.

1.2.1 PDE-based Image Inpainting

Initially, PDEs have widely been used for image denoising tasks [206, 15, 44]. Here, they are employed as diffusion filters which act as smoothing operators. In the simplest case of homogeneous diffusion, it can be shown that the filter is equivalent to a

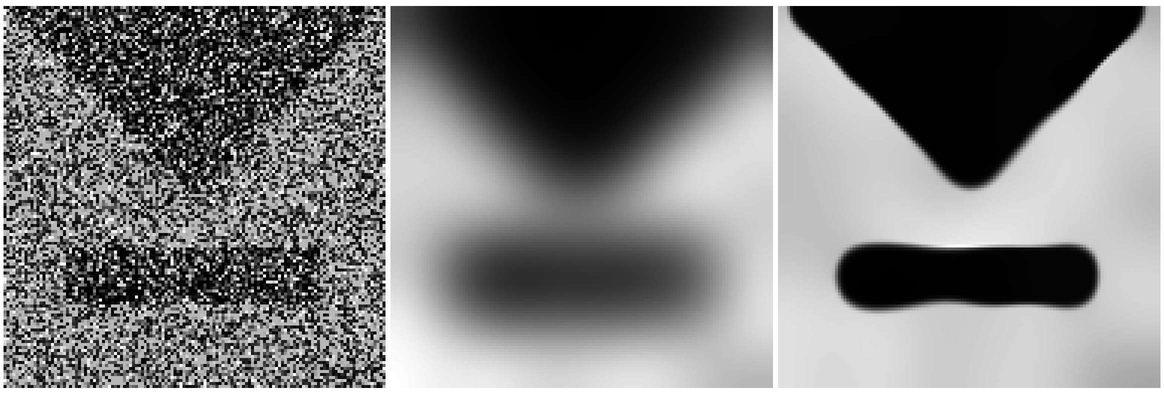

Figure 1.2. Example of PDE-based image denoising (source: [206]). **Left:** Original noisy image. **Centre:** Denoised image with homogeneous diffusion. **Right:** Denoised image with edge enhancing anisotropic diffusion (EED).

Gaussian smoothing. Another common choice is biharmonic diffusion, which relies on a second order Laplacian instead of a first order one. In [204, 205, 206, 207], more complex nonlinear and anisotropic diffusion processes are analysed in the context of image processing. In particular, edge enhancing anisotropic diffusion (EED) or coherence enhancing anisotropic diffusion (CED) have proven to be very powerful for image denoising, as they are able to preserve important image features such as edges or coherent structures.

An example of how such diffusion filters perform in the context of image denoising is shown in Figure 1.2. As one can see, more sophisticated PDEs can have a better denoising performance: While homogeneous diffusion reconstructs the image with a simple isotropic smoothing of the noisy input image, edge enhancing anisotropic diffusion additionally preserves and even enhances important image edges. This is achieved by a cleverly designed PDE, which steers the diffusion along contours, while reducing diffusion across them.

Besides the favourable performance in image denoising, it turned out that PDEs are very powerful in the context of image inpainting, too. The underlying idea has been originated in 1998 by Masnou and Morel [131], while the first major work in this direction has been presented by Bertalmío *et al.* [25]. Over the past years, this concept has been extended and modified in several different ways, see e.g. [43, 88, 196, 26]. The idea behind image inpainting is to recover some missing parts of an image with the help of the surrounding known information. PDEs allow to fill these gaps by diffusing the known tonal data into the unknown regions. Here, tonal values denote the grey or colour values, respectively. The diffusion process can be seen as a filling-in effect, and works follows: While known data is kept fixed by considering it as Dirichlet boundary condition, one solves a suitable partial differential equation to obtain an inpainting solution. Thereby, the used inpainting operator highly influences the result. It is natural to assume homogeneous Neumann boundary conditions at

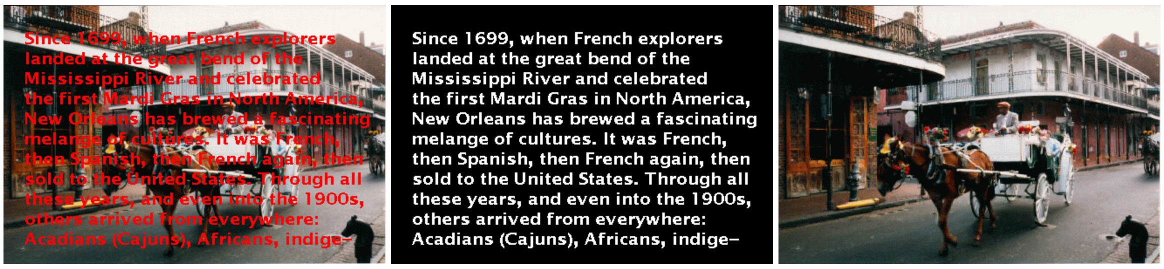
Since 1699, when French explorers landed at the great bend of the Mississippi River and celebrated the first Mardi Gras in North America, New Orleans has brewed a fascinating melange of cultures. It was French, then Spanish, then French again, then sold to the United States. Through all these years, and even into the 1900s, others arrived from everywhere: Acadians (Cajuns), Africans, indige–

Figure 1.3. Example of PDE-based image inpainting (source: [25]). **Left:** Original corrupted image by some font. **Centre:** Inpainting mask: Black pixels denote fixed prescribed data, while missing parts in white are inpainted. **Right:** Result of image inpainting: Font is successfully removed.

the image boundaries. In total, the inpainting process can be considered as solving a mixed boundary value problem.

An example is depicted in Figure 1.3. Here, image inpainting is employed to get rid of some undesired image font. To this end, the known information from the background image is diffused into the unknown region. A so-called inpainting mask indicates missing regions in white. The image in the background denoted by black mask pixels is kept fixed. As one can observe, the missing image is reconstructed very well with the help of PDE-based inpainting.

Based on the promising performance of PDE-based inpainting, it is tempting to carry this concept to an extreme: Instead of just filling in some missing holes, a further step is to reconstruct an image from just a few prescribed tonal values at sparsely sampled locations. As in image inpainting, the missing data is recovered from the given data by solving an appropriate partial differential equation. The known data again provides Dirichlet boundary conditions. This concept has been analysed in detail in [41] for linear partial differential equations, while inpainting with the nonlinear EED is proposed and investigated in [78, 79]. Interestingly, inpainting using the biharmonic operator comes down to thin plate spline interpolation [63].

An example of the power of PDE-based inpainting in such a scenario is shown in Figure 1.4. As we can see, the recovered images are very convincing. Even though it is very hard to guess the image content only from the very sparse data, PDE-based inpainting methods are able to reveal the underlying image. Among the three inpainting operators, EED gives the best reconstruction in terms of visual perception. This is best visible at the contours within the inpainted images. However, this favourable performance goes hand in hand with some drawbacks. More specifically, the more complex nonlinear inpainting process usually leads to a higher computational effort, and there is a need to specify some parameters to obtain good results. On the other hand, homogeneous diffusion inpainting as well as the biharmonic inpainting also lead to very convincing results, and are completely parameter-free and also simpler in spirit due to the linearity.

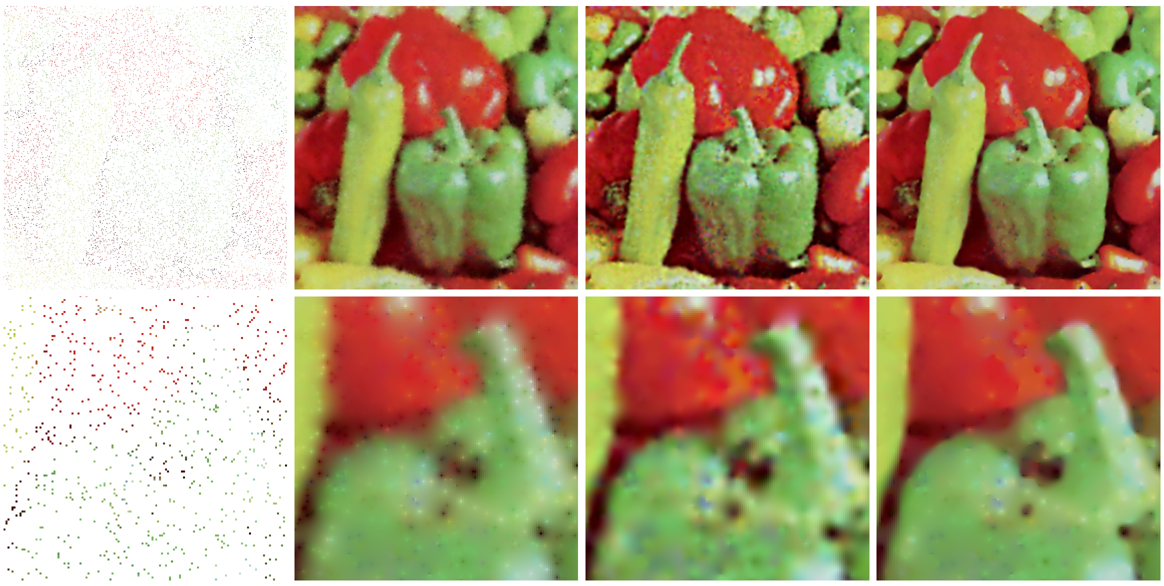

Figure 1.4. Image inpainting from sparse data. **Top left:** Given colour values at randomly sampled sparse mask points. Only 4% of all pixels are selected (image size: 512×512). **Top middle left:** Homogeneous diffusion inpainting. **Top middle right:** Inpainting with biharmonic operator. **Top right:** Reconstruction with EED inpainting. **Bottom row:** Zoom-in.

In general, the promising performance of PDE-based inpainting directly suggests to employ such inpainting approaches for the purpose of image compression. This seems to be a intuitive idea as very little data is necessary to store image information. However, although the basic idea is relatively straightforward, we will see that there are many issues which one has to address in order to get a good compression performance.

Related Methods

The potential of PDEs for image inpainting can also be found within variational approaches in the context of computer vision and image processing. Examples include variational optic flow estimation [94, 142, 37], stereo reconstruction [130, 217] or image registration [18, 137]. In all of these methods, a solution is found by minimising an appropriate energy functional consisting of a data and a smoothness term. The associated Euler-Lagrange equations, which represent a necessary condition for a minimiser of the energy functional, are given as a PDE. The corresponding smoothness part can be interpreted in terms of a filling-in effect by PDE-based image inpainting. Thus, the regularisation given by the smoothness term is responsible to diffuse known information into regions where the data term does not provide useful information. The relation between the regularisation term and diffusion filters has also been extensively studied in [167].

Another prominent field of application of PDE-based inpainting methods is image zooming [127, 106, 20, 22, 13, 209, 162, 34]. Here, the task is to increase the resolution of a part of an image by interpolating the given data. In this task, the prescribed data is typically lying on a regular grid. Hence, in this special case it possible to also employ specifically tailored interpolation strategies. Examples comprise interpolation with bi-linear, bi-cubic or bi-quintic splines, with radial basis functions, with wavelets or sinc-based interpolation [114, 133].

In contrast to data lying on a regular grid, we focus on a more general case of interpolation where the data can be prescribed at arbitrary positions within the image domain. This is also known as scattered data interpolation [76, 146, 14, 71, 61]. In this context, so-called radial basis functions are frequently employed [38, 198]. Some specific instances of such radial basis functions can be interpreted in terms of multidimensional spline interpolations. A prominent example are thin plate splines [63] which can be related to biharmonic inpainting as already noted before. Besides 2D image interpolation, radial basis functions are also widely used in 3D applications, as for example to reconstruct and represent surfaces [76, 40, 58].

Some radial basis functions can be seen as fundamental solutions of specific continuous PDEs, which allows to relate scattered data interpolation using radial basis functions with PDE-based inpainting solutions [38]. However, it is important to mention that these fundamental solutions do not pay attention to any boundary condition by definition. Additionally, such approaches usually do not consider the discrete setting. For these reasons, one has to be careful when establishing a one-to-one connection to solutions which are obtained by solving PDE-based image inpainting problems. As we will see later in Chapter 3 we are able to represent an exact solution of discrete inpainting problems with the help of so-called discrete Green's functions.

Besides tonal information, also different image features can be used to reconstruct an image. For instance, it is is demonstrated in [219] that zero-crossings of an image contain a lot of information and allow to retrieve a very good approximation of the underlying image. In [131], level lines are used to interpolate occluded parts of an image. Considering surface reconstructions in 3D, it is shown that it is possible to reconstruct surfaces only from a sparse set of oriented points [175]. Similar to other methods mentioned before, a higher order variational approach is employed as basis for a whole taxonomy.

Considering the inpainting process itself, the original approach of Masnou and Morel has been modified in several ways. Chan and Shen for example used total variation (TV) regularisation for image inpainting [178]. In [43], they proposed a closely related curvature-driven diffusion (CDD) which is designed to connect contours for non-textured images. In contrast to the classical TV inpainting, CDD more likely connects separate parts with larger distance. In [27], a complex Ginzburg-Landau equation is used to reconstruct higher dimensional data from a sparse subset which allows to reduce the amount of kinks that are visible in the approach of Masnou and Morel.

A common framework for PDE-based inpainting for vector valued data is presented in [196]. A fast image inpainting method based on coherence transport is proposed in [26]. Another anisotropic image inpainting scheme is introduced in [88].

Most of the aforementioned methods try to fill-in missing information by a suitable smoothing operator. Although important image features such as edges can be restored, e.g. with anisotropy or nonlinearity, the reconstruction of textured image regions often remains an issue for methods relying on partial differential equations. In this context, Efros and Leung were the first to introduce exemplar-based inpainting which allows to propagate a small sample texture over a large image [66]. Later, Facciolo *et al.* extended this idea for the purpose of texture synthesis from sparse data [69]. Very recently, Peter *et al.* combined the advantages of such exemplar-based inpainting methods and traditional PDE-based inpainting methods [155]. This allows to reconstruct both smooth regions as well as textured regions at the same time. Moreover, there are also first approaches for patch-based inpainting, which is designed to reconstruct textures well [176]. Further investigations considering texture inpainting are presented in [223].

PDE-based inpainting methods are also useful when it comes to the colourisation of grey-scale images. Based on the work of Levin [116], it could be shown that only a few sparsely sampled given colour strokes allow to fill a complete grey-value image with colour [105, 152]. Thereby, the idea is to steer the diffusion process with the help of the existing grey-valued image. This allows to interpolate the information of the colour scribbles, while preserving important image edges within the colour channels. Colourisation methods based on related variational methods also exist, see for example [103].

A sparse set of significant points can also be employed for reconstructing an image from so-called top points which represent feature points in a Gaussian scale space [100, 104]. Besides, the potential of other feature points and their derivatives - the local jet - to reconstruct and describe images is presented in [120].

As one can see from all those various approaches, PDE-based image inpainting is a very popular and flexible tool with very good reconstruction capabilities.

1.2.2 From Inpainting to Compression

As it has been demonstrated in Section 1.2.1, PDE-based inpainting methods also offer a high potential of recovering an image from only a small amount of prescribed data. This property directly suggests to exploit this potential for compression methods, leading to the following basic idea: In the encoding phase, the position of the individual mask points as well as the corresponding tonal values have to be stored. At the decoding side, the decompressed image is then obtained by inpainting the stored data.

Although this seems to be a straightforward approach, it turns out that obtaining an efficient image compression method that competes or even outperforms standard approaches is by far not trivial. In this context, an important point is that the compression algorithm can be freely designed. This means that instead of facing a predefined setting, there are many degrees of freedom in the design decision. The most important aspects are discussed in the following.

Selection of inpainting operator

As one can observe from Section 1.2.1, the result of a PDE-based inpainting highly depends on the differential operator. This leads to the question of which operator is suited best for the purpose of image compression. In [78], it is demonstrated for a randomly sampled mask that EED inpainting usually gives better reconstructions compared to a harmonic and biharmonic inpainting. This is confirmed by experiments in [30] and [170]. Even for optimised interpolation data, it turns out that EED inpainting is the more favourable choice as we will also see in Chapter 5. The reason for this is the ability of EED to reconstruct edges and texture in a better way. However, one also has to keep in mind that the complexity of an operator also plays an important role when it comes to the performance. Thus, one may not underestimate the potential of linear inpainting operators. Moreover, when it comes to image compression, a fair comparison of different operators is only possible when comparing the final compression codec, i.e. when the maximum compression potential of the operator is exhausted. This includes all inherent optimisations as well as the storage of the sparse inpainting data. For the aforementioned reasons, it is not possible to find an ad-hoc answer to the question, and we will see that also linear operators can lead to comparable results as EED-based approaches.

Selection of inpainting data

Instead of prescribing tonal values at given locations that need to be filled in, it is possible to select this data freely in a compression context. One can observe that the quality of the reconstruction highly depends on the prescribed image information. For instance, we will see that it makes sense to optimise the tonal data in the encoding phase. This means that instead of the tonal values from the original image, modified value are specified at the mask locations. Even though this introduces an error at the respective mask locations themselves, the overall reconstruction error becomes smaller. Already in 2008, this idea has been realised by a first greedy approach [79]. However, we will see that a nice theoretical framework can be established for finding optimised values. Besides the tonal data itself, its locations also have a high influence on the reconstruction. For example, a randomly sampled mask usually gives much worse results compared to a cleverly selected set of points. As we will see later, this optimisation task is far from being trivial as it comes down to a non-convex optimisation problem. One can also consider it from a combinatorial point of view. For example, selecting 5% of the pixels of an image of size 256×256 leads to the following number of possible combinations:

$$\binom{65536}{3277} \approx 1.72 \cdot 10^{5648}.$$

Obviously, trying out all possible combinations is not feasible. For this reason, a clever selection of the mask points has to be employed. For the continuous setting, Belhachmi *et al.* [23] showed for homogeneous diffusion inpainting that the optimal position of mask points has to be at locations where the magnitude of the image Laplacian is large. In the paper, the authors proposed to employ a dithering to obtain a bridge to the discrete setting. The resulting mask already yielded very promising results. However, we will see that this is not the end of the road. It is apparent that in the ideal case, the positional and tonal information is optimised in a joint fashion to find the best possible combination. However, this requires to solve a complex optimisation problem. In this context, an optimal control framework was proposed for finding an optimised mask [92]. We will discuss the issue of selecting inpainting data in more detail within this work.

Quantisation

If one does not restrict the tonal values, they are typically wide spread over a large range. In particular after optimising them, one typically obtains real valued data. It is clear that storing these values in a high precision would prevent from an efficient compression algorithm due to the high memory requirement. Thus, in a compression context, it makes sense to quantise the tonal values such that they are only allowed to be within a small set of possible values. Although this reduces the precision of stored tonal data and consequently leads to a slightly degraded inpainting result, the hope is that the loss of quality is compensated by a much lower coding cost of the tonal values. In particular, the human visual system is only capable of differentiating around 40 different grey scales. Thus, it seems natural to store even less than 256 different grey values that fit in one byte. By reducing the storage costs of the tonal data, one can even think of choosing more mask points in reverse. Usually, a quantisation parameter is employed to steer the inherent trade-off between quality and coding efficiency.

An interesting aspect is that even if only very limited tonal values are stored, the advantage of PDE-based inpainting is that the reconstructed values at non-mask locations can take arbitrary real-valued numbers. Thus, especially when the encoded quantised values are chosen cleverly, the reconstruction can be of a very good quality without showing typical quantisation artefacts as it is the case in JPEG or JPEG2000.

The concept of quantisation has already been proposed in the beginning of 1960 by Max [132] and is a main ingredient of compression methods. Also JPEG and JPEG2000 rely on a quantisation based on the following observation. Humans are more sensitive to lower frequencies, while focusing less on higher frequencies. Thus, the coefficients belonging to different frequencies are stored with varying precision:

The coefficients belonging to higher frequencies are quantised higher and therefore stored with a lower accuracy. This allows to give lower frequencies more importance in the encoding phase. Obviously, incorporating quantisation in JPEG and JPEG2000 led to very successful compression algorithms.

Entropy coding

Clearly, the selected mask points and quantised tonal values have to be stored in a compact file representation. A naïve way to achieve this would be to simply write this data byte-wise into a file. Of course, this procedure is not the optimal choice. One has to find a way to store the inpainting data more efficiently. This is where entropy coders enter the game. They are frequently used to efficiently store data in a lossless manner, and is achieved by reducing the redundancy in the data. Prominent examples are Huffman coding [96], adaptive and static arithmetic coding [160, 161] or PAQ [123]. As demonstrated in [170], such entropy coders can directly be applied to the bit stream given by the mask point locations and the tonal data. The more sophisticated PAQ consists of a context mixing model which employs a neural network to subsequently predict the individual bits in the stream. This way, it achieves a better compression performance compared to the conceptually simpler Huffman or arithmetic coding [170].

As the mask alone can be considered as a binary image, where a 1 represents a mask point, there are also other approaches to compress the mask. One possibility is represented by block coding schemes, which decompose the image into small blocks and thereby benefit from the localisation [112, 220, 77]. Another promising approach is JBIG [102] which has been originally introduced for the purpose of fax transmission. By considering the context of a pixel given by the configuration of surrounding points, it tries to predict the value at a respective image position. Subsequently, an arithmetic coding is employed for encoding. The successor of JBIG, namely JBIG2, additionally introduces a dictionary to capture more likely structures that typically arise in halftoned images or images containing text [95]. Whenever the dictionary does not help, it employs JBIG as a backup. Typically, the masks used for inpainting purposes do not reveal such structures, which is why JBIG2 is not supposed to be advantageous. Besides JBIG and JBIG2, there is also DjVu [28], which decomposes the image into a binary foreground image and a background part. The background, typically consisting of textured information, is then compressed with the help of a wavelet transform, while the foreground is encoded with a variant of JBIG2.

Trade-off between data optimality and coding efficiency

On the one hand, one is interested in finding optimal data that allows the best possible reconstruction. However, there is a good chance that the selected optimal data is very hard to be stored efficiently due to its irregular structure. On the other hand, one could think of selecting a fixed mask pattern, as for example a mask deduced from a regular grid structure. While such a mask can be stored with no overhead, the resulting inpainting is far from being optimal. Of course, there are countless of variants in between these extreme scenarios. To handle this trade-off, it might be a good idea to choose the data while having in mind the associated coding costs at the same time. One promising and thoroughly investigated strategy in this direction is discussed in [78, 79, 172, 170]. Here, the mask points are selected with respect to a recursive subdivision of the image, which is locally adapted to the image. The advantage of specifying a mask in this way is that the it can be fully represented by a binary tree. This tree structure is very efficient to store, especially in combination with one of the before-mentioned entropy coders, and flexible enough to adapt to the image structure. As it turns out, it is hard to find more efficient entropy coders for the purpose of storing such binary trees [166].

Selecting the Optimal Features

Apparently, there are many degrees of freedom when designing a compression algorithm based on PDE-based inpainting. This can be considered both as an advantage and a drawback at the same time. On the positive side, this flexibility yields the opportunity to improve and perfect the codec. On the negative side, the flexibility inherently comes with a big burden: Finding the optimal combination of the individual ingredients and tuning them to the extreme can be very complex. Ideally one would have to optimise all these features in a joint fashion to find the best possible reconstruction in the end. However, this is very hard due to the large number of unknowns which influence each other as well as the resulting reconstruction. For example, the entropy coder highly depends on the chosen mask, while the mask and the tonal data is dependent on the quantisation. Moreover, the inpainting depends on the chosen mask and the tonal values.

To circumvent this issue, one often splits the problem into several sub-problems. To this end, the individual components are separated into groups and then optimised one after another. For instance, one typically first selects the inpainting data, and tries to find a good quantisation parameter afterwards. Often, the task of finding the inpainting data is split further into first finding a mask, and a subsequent optimisation of the tonal data. The entropy coder is typically selected in the end. It is clear that this strategy does not yield the globally best compression performance. However, by doing so, the complexity of the optimisation problem can be reduced to a manageable amount, while hopefully being close to the optimum.

1.2.3 Existing Methods in PDE-based Compression

In the following, an overview of existing methods in the context of PDE-based image compression will be presented.

PDEs as Pre- or Postprocessing of Existing Codecs

In the context of compression, PDE-based methods and related variational methods have been widely used as a preprocessing step for coding images and videos [75, 74, 190, 45, 197, 109], or as a postprocessing to remove coding artifacts [74, 214, 215, 85, 141, 64, 12, 33]. In contrast to these approaches, our focus is on methods where PDEs represent the core part of compression. In [158], PDE-based inpainting is used to reconstruct missing blocks in JPEG due to erroneous wireless transmission.

PDEs within Existing Codecs

Some approaches integrate PDE-based inpainting ideas into standard transform-based codecs [158, 202, 121, 213, 216, 49]. Here, the objective is to benefit from the ability of PDE-based inpainting methods to reconstruct certain image features. For example, EED inpainting is useful to reconstruct sharp edges, while exemplar-based inpainting such as the approach of Efros and Leung [66] or structure inpainting [158] can be employed to reconstruct specific textured regions. Since transform-based methods usually only perform well in highly textured regions, the compression result can be improved by incorporating inpainting approaches within the remaining regions. However, due to a subsequent patching of the existing codecs, it is clear that the full potential of the individual ingredients of the compression methods is not exploited.

Codecs based on EED Inpainting

In the context of PDE-based image compression, EED-based inpainting is employed as key ingredient in many approaches. In particular for general purpose image compression applications, this inpainting operator is usually the standard choice. The reason for this choice is due to its advantageous performance: As mentioned in Section 1.2.1, experiments have shown that EED-based inpainting has a higher potential to reconstruct an image from scattered data compared to a harmonic and biharmonic inpainting [78]. This can be explained by the potential of anisotropic diffusion to reconstruct edges and contours. In the context of inpainting, more extensive comparisons to other operators are presented in [170]. Here, it is shown that EED inpainting is the preferable method for reconstructing edges, contours and even shapes.

For these reasons, already the very first PDE-based image compression codec proposed by Galić *et al.* in 2005 employs EED-based inpainting [78]. This approach was able to beat the quality of the widely used JPEG standard. To do so, an inpainting mask is selected as vertices of an adaptive triangular subdivision of the image domain. This triangulation can be stored efficiently with the help of a binary tree structure. The splitting decision is made on the local error, i.e. more points are recursively added at regions where the difference between the linear interpolation within a triangle and the original image is large. For decoding, the reconstruction is obtained with the help of EED inpainting.

An improved version of [78] has been presented in 2008 [79]. Here, the splitting decision is based on the difference between the original image and the current reconstruction obtained by EED inpainting. This improves the mask selection procedure tremendously on the cost of an increased encoding time, because the complete inpainting solution has to be computed after each point insertion. By additionally incorporating a specific quantisation strategy and an error threshold adaption for the splitting process, the final compression algorithm already comes close to the compression capability of JPEG2000.

An even more advanced version of this codec has been presented by Schmaltz *et al.* in [172]. This so-called R-EED method uses an adaptive rectangular subdivision instead of a triangular one. Also, an interleaved optimisation of the contrast parameter of EED inpainting and the quantised grey values is applied. With the help of a so-called interpolation swapping and another entropy coding for storing the data, it is possible to further improve the compression performance. In the end, the R-EED codec is able to outperform JPEG2000. More extensive experiments and comparisons to other inpainting operators are presented in [170]. As a result, the authors showed that EED turns is the most favourable one. Moreover, extensions of R-EED for shape coding and 3D data compression are proposed in the paper, based upon the work in [98] as well as [10, 151]. Further, adaptations of the R-EED codec to a progressive mode as well as region of interest coding are proposed in [153], as an extension to [169]. A probabilistic approach to subdivision trees in the context of PDE-based image compression is presented in [143].

There also exist other applications which base on the R-EED method. For instance, a video compression codec is presented in [171] in which concepts like 3D pose tracking and electrostatic halftoning are successfully combined with PDE-based compression methods. Another example is the application of R-EED in the context of steganography and in hiding high resolution images in low resolution versions of themselves [126, 147]. EED inpainting can also be a powerful tool for the purpose of shape coding. This is demonstrated in [98, 183], where the shapes are encoded with the help of quadrupoles and nonapoles, respectively. As an extension of bipoles, such features are able to represent the directional information of edges which can be propagated over the whole image by EED inpainting.

When it comes to colour images, all the aforementioned compression codecs relying on EED inpainting treat the three RGB colour channels in a straightforward way. However, it has been shown that it can pay off to perform the compression and in particular the inpainting part in another colour space such as YCbCr [154]. This allows to adapt the coding to the human visual system by privileging the luma channel: First, the luma channel Y is encoded in a higher quality with R-EED. Afterwards, for reconstructing the chroma channels Cb and Cr, the inpainting is performed by steering the diffusion by the luma channel. A more detailed description and analysis of this colourisation idea is presented in [152].

Despite the success of R-EED and its variants, the compression of highly textured images remains a challenge for PDE-based methods. To cope with this problem, a hybrid image compression method has been proposed in [155]. Here, the R-EED codec is successfully combined with the exemplar-based inpainting of Facciolo *et al.* [69]. This way, the benefits of EED inpainting to recover smooth regions along with sharp edges are combined with the possibility to recover textured regions. In the end, the new approach allows to compete even with transform-based methods for textured images.

Besides the qualitative assessment of these PDE-based compression approaches, it is also important to consider their potential run-time performance. In this context, Köstler *et al.* have shown that it is possible to perform the compression method presented in [78] in real-time on a Playstation 3 [110]. Further, it is demonstrated in [21, 153] that the more advanced R-EED codec of Schmaltz *et al.* [170] can be used for real-time video decoding on a modern computer with graphics card, too. To obtain the inpainting solution in an efficient way, a so-called *fast explicit diffusion* (FED) scheme [208] is employed as efficient numerical solver, along with speed-up strategies such as a clever initialisation of the inpainting solution. More details on the FED method will be given later.

As one can observe in all these approaches, EED inpainting is the natural choice for general purpose coding due to its favourable reconstruction potential. However, all these approaches that rely on EED suffer from the fact that the crucial optimisation of the inpainting data in the encoding part is very time consuming. Besides the mask locations and the tonal values, also the parameters of the inpainting operator have to be selected in a good way. As the reconstruction quality highly depends on these factors, this part is very important. However, due to the non-transparent dependency of these parameters and the resulting inpainting solution, this procedure is highly non-trivial as well as time-consuming.

Codecs based on Linear Inpainting Operators

Some compression codecs rely on linear inpainting processes like harmonic and biharmonic inpainting instead of EED inpainting. Even though they could not compete with standard approaches on general images so far, very good results could be obtained for some designated compression applications. Thereby, the images typically exhibit certain properties that allow for a good reconstruction with a linear inpainting procedure.

As an example of such a designated image compression algorithm, Mainberger *et al.* have presented an edge-based compression codec which is very powerful for compressing cartoon-like images [125]. The underlying idea of this work is that edges of such images contain the most important information for the visual perception. Although this assumption is not necessarily the optimal one for general images, it holds true for cartoon-like images, since they are characterised by piecewise constant regions. An example of such an image is depicted in Figure 1.5 on the left. The encoding procedure of the approach in [125] consists of two parts: After first extracting the edge

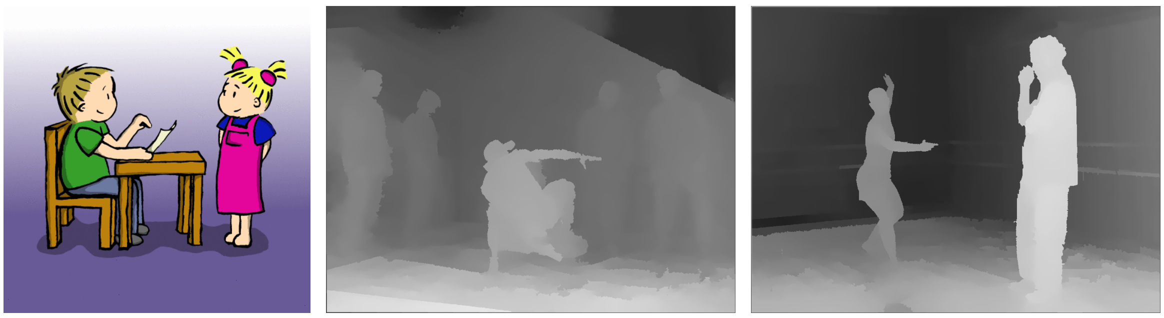

Figure 1.5. Left: Example cartoon-like image *comic* (size: 512×512). **Centre and Right:** Example depth maps *breakdancers* and *ballet* (size: 1024×768, source: [224])

locations, the tonal values are encoded along both sides of these contours in a second step. To reconstruct the image from this data in the decoding part, homogeneous diffusion inpainting is employed. The harmonic operator is sufficient to reconstruct the constant regions within the image. It could be shown that this approach is able to outperform standard codecs such as JPEG or even JPEG2000 for images of this specific type.

Another example of designated image compression algorithms based on linear inpainting methods is the compression of depth maps, see for example [80, 118]. Depth maps become more and more important as they are part of the multi-view plus depth (MVD) format which is used for 3D television. It is clear that one is also interested in efficient compression methods designated for such depth images. Example depth maps are shown in Figure 1.5. The grey values within such a depth map represent the distance to the camera centre, i.e. bright pixels denote close objects, while far away objects are depicted in darker grey tones. As one can observe, such depth maps are characterised by piecewise smooth regions. The proposed methods [80, 118] base on the work in [125], and adapt it to the needs of depth map compression. Due to the fact that the edges in depth images are usually much sharper than in comic images, they use a smaller gap between the stored grey values along the edges. Besides, they employ another edge extraction method, different entropy coders as well as an additional regular mask to capture smooth variations withing the regions. Homogeneous diffusion inpainting yields the reconstructed depth map at the decoding stage. The final compression methods are able to compete with JPEG2000. However, the compression efficiency of the more advanced HEVC algorithm could not be attained. As we will see in this work, PDE-based compression methods are capable to compete even with HEVC by cleverly tuning the individual ingredients. First work in this direction has been performed in [157].

Considering the compression of 3D data with the help of homogeneous diffusion inpainting, there are also first results available [122].

Variant Approaches

Besides the aforementioned methods, there exist many different variant approaches in the literature. They mainly differ by different design decisions. The most important ones are presented below.

Different inpainting operators Apart from EED-based inpainting or inpainting based on linear operators, also other operators have been considered for image compression tasks. In [41], an absolute minimal Lipschitz extension is considered. Total variation (TV) inpainting is also employed in certain applications. For instance, in [46], coefficients of a wavelet decomposition are modified with the help of TV regularisation to alleviate oscillatory artefacts. Moinard *et al.* [138] employed TV inpainting to predict DCT coefficients within a codec which relies on JPEG. Image compression with the help osmosis is investigated in [89].

Different features Instead of reconstructing an image from a prescribed set of points, there are also other image compression approaches which try to recover an image from specific features. These features are often perceptually motivated and try to reflect important characteristics of the human visual system. The resulting methods belong to the class of so-called second-generation coding [111].

One major class of such features are contours and edges which play an important role for the human perception of arbitrary images. In this context, Werner has already studied the significance of edges within images in 1935 [210]. Further, Elder has demonstrated that edges comprise the most relevant information of an image and are sufficient for a good visual impression [67]. For these reasons, it seems natural to focus on these features for compressing images. There exist prominent examples where the image is successfully encoded and reconstructed from the contour information [39, 97, 16, 60]. In [97] it is demonstrated that edges on multiple scales contain the most important information of an image.

Despite these first results, it is not as straightforward as it seems on first glance to obtain a state-of-the-art compression algorithm relying on edge information. As already mentioned, there exist compression approaches which perform very well for cartoon-like images or depth maps [125, 80, 118]. When it comes to general purpose compression, there also exist some publications which rely on edges or contours [9, 60, 113, 121, 211, 19, 221, 49, 84, 36]. Yet, all these approaches still lack compression efficiency, mainly because selecting only edges as inpainting data may not always be sufficient. This hypothesis is supported by the findings in the paper of Belhachmi *et al.* [23], which basically states that mask points should be more likely at positions with a large magnitude of the Laplacian. Although edges locations count as such locations, there might also be other important locations.

Another feature that has been used in a PDE-based compression context are corners [222]. Recently, a promising alternative to storing some tonal values is to encode sparse derivative information instead. The reconstructed image is obtained by inpainting the derivatives first and subsequently integrating the resulting derivatives. After first attempts in [174] and theoretical considerations in [36], a compression framework was recently established by Schneider *et al.* [8].

Localisation of inpainting A large number of compression approaches employ a localisation of the reconstruction procedure by subdividing the image domain into triangles or rectangles [65, 187, 188, 62, 59, 107, 115, 108, 29, 182, 54, 117]. Often, the subdivision is obtained with an adaptive triangulation or tree structures such as B-trees or quadtrees. In most of the methods, the reconstruction within each region is obtained with the help of a linear polynomial interpolation. The advantage of these approaches is that the reconstruction can be computed efficiently by exploiting the small spatial support as well as the independence of the individual regions. Interestingly, it could be shown that these methods can be competitive with JPEG2000 for images containing no texture [59]. By considering the tonal values at the region boundaries as Dirichlet boundary data, one obtains a straightforward relationship between the reconstruction and a localised PDE-based inpainting. In Chapter 6, we will also employ such a strategy for designing efficient codecs. Moreover, there is a close connection of these localisation methods to finite element methods which aim at approximating solutions to continuous PDEs by describing them within small regions [101, 53].

Different data Besides usual images, PDE-based compression methods are also applied to variant data. For instance, a compression algorithm for triangulated surfaces is proposed in [17]. Here, the authors rely on a geometric diffusion equation for the purpose of scattered data interpolation. Besides, PDEs are used for the compression of digital elevation maps [181, 70, 212]. More recently, it is demonstrated that such elevation maps can be encoded efficiently by storing the level lines and reconstructing the map with the help of diffusion inpainting [173].

1.3 Objectives

In this thesis, we want to investigate the true potential of linear PDEs in an image compression context. In particular, we mainly focus on the harmonic and biharmonic inpainting operators, but also consider higher order or nonlinear operators whenever it fits. By carefully examining all individual ingredients of the compression algorithm, we try to tune the coding performance to an extreme. While linear PDEs are said to be less powerful than nonlinear PDEs, one goal of this thesis is to show that they can also play a major role in practical applications, even when it comes to the compression of general images.

Compared to its nonlinear counterpart, linear inpainting has several advantages. First of all, linear inpainting operators are completely parameter-free which naturally avoids any need to choose or optimise these parameters. Additionally, the corresponding conceptually simpler PDEs typically require less computational and implementation effort. Moreover, we will see that specific theoretical concepts and the resulting algorithmic advantages are only valid for linear operators. All these aspects show that it is worth to have a closer look at compression with linear PDEs. As we will see, a few methods for the linear operators can be transferred to the nonlinear case as well.

On the way of designing the compression codecs, a fundamental and vital part of this work will be to thoroughly analyse and investigate the inpainting process itself. Besides obtaining a much better understanding, this will furthermore enable us to derive efficient and well-performing strategies for different parts of the compression algorithm. For example, we will see that discrete Green's functions allow for a reinterpretation of the inpainting problem in terms of sparsity, which in turn offers advanced spatial and tonal optimisation strategies.

By combining these sophisticated data optimisation strategies with efficient encoding and decoding strategies, the final hope is to come up with powerful compression codecs which can compete with standard approaches like JPEG, JPEG2000 or even HEVC. Besides the qualitative performance, this also means that we aim at fast methods with comparable decoding times as the standard algorithms.

1.4 Outline

After this introductory chapter, we will have a close look at the standard PDE-based image inpainting problem in Chapter 2. Besides giving a general overview, different properties of the inpainting solution and the inpainting problem itself are presented and analysed. Afterwards in Chapter 3, the theory of discrete Green's functions is presented in detail. In this context, fast computation techniques including the efficient generation of discrete Green's functions are shown. Then, it is demonstrated that these concepts play a major role in the context of PDE-based image inpainting. In particular, we will see that it is possible to reformulate the inpainting solution in terms of the discrete Green's functions. This in turn will offer novel theoretical insights as well as efficient alternative computation strategies.

With these fundamental concepts of the inpainting process, we will address the issue of finding optimal data. First, we focus on optimising tonal data at prescribed mask locations in Chapter 4, while seeking optimal mask locations afterwards in Chapter 5. For both tasks, different strategies are presented, discussed and compared. In particular, we will also investigate the benefit of the new inpainting formulation with discrete Green's functions to accelerate or improve the optimisation procedure.

Based on the gained insights, three different compression frameworks are then presented and discussed in Chapter 6. First, we will consider a designated compression approach for depth images. Subsequently, a compression method for general images is presented which relies on the best possible data optimisation techniques. While the first two methods mainly focus on the qualitative performance, the third compression algorithm will be tuned with respect to both quality and decoding runtime. For all three proposed compression methods, experiments will be performed and comparisons with standard approaches will be presented.

Finally, a summary and a conclusion is given in Chapter 7, along with an outlook on future work. A bibliography can be found in the end.

Chapter 2

PDE-based Inpainting

As mentioned in Chapter 1, inpainting with partial differential equations is a very powerful tool to interpolate sparse data, and is hence employed in many image compression applications. Also in this thesis, PDE-based inpainting plays a central role. Thus, it is important to study it in more detail. To this end, we will first introduce the continuous formulation of the inpainting problems. As we will see, they originate from the continuous theory of diffusion processes which can be described by partial differential equations. More in-depth information on diffusion processes in the context of image processing is presented in [206]. Moreover, we also focus on the discrete formulations of the inpainting problem. They are very important as images typically reveal a sampling on an equispaced rectangular grid structure. Hence, will present and analyse the discrete problem in more detail to obtain a foundation for all further considerations within this thesis.

The outline of this chapter is as follows. We start by recapitulating the physical background of diffusion processes in Section 2.1, and consider the continuous inpainting formulation afterwards in Section 2.2. The discrete inpainting problem is then presented in Section 2.3. Example results are shown in Section 2.4. After analysing the inpainting problem in Section 2.5, we consider an extension of the inpainting to non-binary masks in Section 2.6.

2.1 Physical Background of Diffusion Processes

Diffusion is a physical process that equilibrates concentration differences within some medium. In the context of image processing, the concentration is usually given by the grey or colour values, respectively. The diffusion process can be formulated mathematically as follows. Assume there exists a concentration gradient $\boldsymbol{\nabla} u$, then it creates a flux $\boldsymbol{j}$ to compensate for this disequilibrium according to *Fick's law*:

$$\boldsymbol{j} = -\boldsymbol{D}\,\boldsymbol{\nabla} u. \tag{2.1}$$

Here, $\boldsymbol{D}$ is the *diffusion tensor*, a positive definite symmetric matrix that describes the relation between the flux and the gradient. The diffusion tensor controls the direction of the diffusion process. In an isotropic setting for example, $\boldsymbol{j}$ is parallel to $\boldsymbol{\nabla} u$. In an *anisotropic* case, the flux does not necessarily point in the same direction as $\boldsymbol{\nabla} u$.

During the diffusion process, no existing mass is supposed to be destroyed nor should there be any newly introduced mass. This property is expressed by the *continuity equation*:

$$\partial_t u = -\operatorname{div} \boldsymbol{j} \tag{2.2}$$

The partial derivative $\partial_t u$ denotes the derivative of u with respect to the time t. Combining Fick's law and the continuity equation leads to the *diffusion equation*:

$$\partial_t u = \operatorname{div}\left(\boldsymbol{D}\, \boldsymbol{\nabla} u\right). \tag{2.3}$$

Naturally, there are also extensions and variants of this diffusion equation, such as higher order diffusion schemes. Most of them can be cast into the following general formulation of a parabolic partial differential equation:

$$\partial_t u = Lu. \tag{2.4}$$

In this PDE, the differential operator L describes how the diffusion behaves: Besides isotropic and anisotropic diffusion processes, there is also a distinction between *linear* and *nonlinear* processes. In the nonlinear case, the operator depends on the evolving image, i.e. L can change over time. On the other hand, the differential operator is fixed when one talks about linear diffusion.

For most applications, one also needs to impose some suited boundary conditions and to specify an initial state u at time $t = 0$. In image processing for instance, one usually selects the original image as initial condition and selects homogeneous Neumann boundary conditions.

2.2 Continuous Inpainting Formulation

The aforementioned diffusion schemes are very powerful for performing an inpainting. Given some prescribed tonal values at specified mask regions, the idea is to smoothly fill-in unknown parts of the image by diffusing the given data into the unknown areas. Thereby, one is seeking the steady-state solution of the diffusion process by letting $t \to \infty$. Usually, such a steady-state solution of an ordinary diffusion process is rather unspectacular. For example, a constant flat image is often the result when imposing homogeneous Neumann boundary conditions on the image boundaries. However, by performing the diffusion process only in the unknown image parts while keeping the remaining pixels fixed, the resulting inpainting solution is by far not trivial. Here,

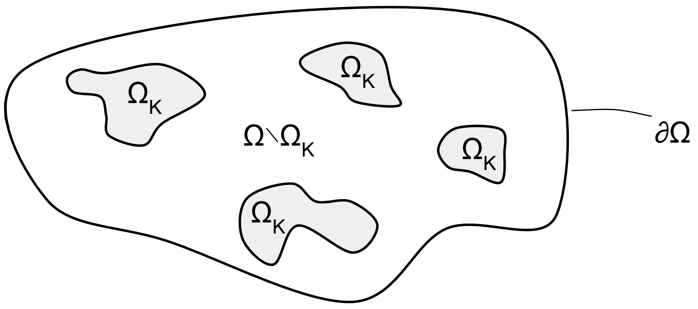

Figure 2.1. Sketch of the inpainting problem setup with mixed boundary conditions: Within the domain $\Omega \subset \mathbb{R}^2$, the tonal values are prescribed at the mask regions Ω_K. The inpainting solution is sought in the unknown region $\Omega \setminus \Omega_K$, while assuming some boundary conditions at the domain boundaries $\partial\Omega$.

the prescribed tonal values are typically given at some mask locations, which can be interpreted as Dirichlet boundary conditions.

To obtain a more formal description of PDE-based image inpainting, let $f : \Omega \to \mathbb{R}$ be a grey-scale image within some bounded domain $\Omega \subset \mathbb{R}^2$. For a location $\boldsymbol{x} \in \Omega$, the grey-value of the image is given by $f(\boldsymbol{x})$. Assume that the image is only known at a specified closed and non-empty set $\Omega_K \subset \Omega$ denoting the mask regions. A sample sketch of such a scenario is depicted in Figure 2.1. The inpainting u is then given as the solution of the following mixed boundary problem:

$$Lu = 0 \qquad \text{on } \Omega \setminus \Omega_K, \tag{2.5}$$

$$u = f \qquad \text{on } \Omega_K, \tag{2.6}$$

$$\partial_n u = 0 \qquad \text{on } \partial\Omega. \tag{2.7}$$

Thereby, ∂_n denotes the derivative normal to the image boundaries. The elliptic differential equation (2.5) gives the steady-state solution of the diffusion process (2.4). The interpolation condition is obtained by enforcing Dirichlet boundary conditions in Equation (2.6). Additionally, homogeneous Neumann (reflecting) boundary conditions at the image domain boundaries are enforced by (2.7).

In this thesis, we mainly focus on the following linear inpainting operators L. First of all, we consider $L = -\Delta$, the Laplace operator, i.e. Equation (2.5) becomes the well-known *Laplace equation.* With this choice, one obtains the so-called homogeneous diffusion inpainting or harmonic inpainting. Despite its conceptual simplicity, harmonic inpainting has proven to be very powerful and to have nice properties such as obeying the maximum-minimum-principle. Its inpainting result is characterised by its smoothness within the interpolated regions. However, at the mask locations, harmonic inpainting suffers from its characteristic peaks or singularities. We will see in Chapter 3 that this behaviour can easily be explained with the help of discrete

Green's functions. Another choice of the inpainting operator will be the biharmonic operator, i.e. $L = \Delta^2$. This higher order operator gives smoother interpolations on the expense of yielding over- and under-shoots, see Section 2.5. It has a close relation to thin plate spline interpolation [63].

Clearly, one can also think of using any degree n of smoothness with an arbitrary power p of the Laplace operator, i.e. $L = (-\Delta)^p$. The advantage of these linear operators is that they are completely parameter-free and lead to a linear system of equations which makes them easier to handle compared to nonlinear ones. Moreover, they are isotropic and thus propagate the information equally in all directions. In particular, they do not depend on the evolving image.

Besides the linear inpainting operators, also nonlinear inpainting with edge-enhancing anisotropic diffusion (EED) [204] will be a topic in this thesis. Here, the differential operator is given by $Lu = \operatorname{div}(\boldsymbol{D}\,\boldsymbol{\nabla} u)$. The anisotropic diffusion tensor $\boldsymbol{D} := \mu_1 \boldsymbol{v}_1 \boldsymbol{v}_1^\top + \mu_2 \boldsymbol{v}_2 \boldsymbol{v}_2^\top$ is designed to adapt to the local image structure of the evolving image u in the following way. The first eigenvector $\boldsymbol{v}_1$ of $\boldsymbol{D}$ is defined to be parallel to the gradient of u, while the second eigenvector $\boldsymbol{v}_2$ is perpendicular to it. This means that $\boldsymbol{v}_1$ points across image edges, and $\boldsymbol{v}_2$ along edges. The corresponding eigenvalues are chosen such that the diffusion process does not only inhibit diffusion across edges, but even enhances edges by smoothing along them. Thus, the second eigenvalue is set to $\mu_2 := 1$, and the first eigenvalue μ_1 steers the amount of diffusion with the help of a fuzzy edge detector given by the Charbonnier diffusivity [48]:

$$\mu_1 := \left(1 + \frac{|\boldsymbol{\nabla} u_\sigma|^2}{\lambda^2}\right)^{-\frac{1}{2}}. \tag{2.8}$$

Here, $u_\sigma := K_\sigma * u$ denotes a smoothed version of u by convolving it with a Gaussian kernel k_σ of standard deviation σ, and λ is a contrast parameter.

By design, EED inpainting is a nonlinear process due to the inherit dependency of the differential operator on the evolving image. Furthermore, there are two parameters σ and λ which influence the final inpainting result and thus have to be determined. Those two main issues slightly complicate some tasks such as finding the inpainting solution, performing a tonal optimisation or analytically describing the inpainting result. Nevertheless, one usually obtains visually more pleasant reconstructions containing sharp edges and without singularities at the mask locations in contrast to homogeneous diffusion inpainting [170]. Note that both the harmonic as well as the biharmonic diffusion inpainting involve linear self-adjoint differential operators. This fact will become important in Chapter 3.

When it comes to colour images, the tonal data of an image $f : \Omega \to \mathbb{R}^3$ is given at the mask locations in the form of vector-valued data. Typically, the three channels represent the RGB colour information. One straightforward strategy to obtain the inpainting solution is to apply the inpainting independently for each channel. This yields the individual channels of the inpainting solution. For EED inpainting, the structure tensor is usually adapted such that it relies on the edges of the combined colour channels.

In the past, other inpainting strategies for colour valued data have been proposed. For instance, in [152], the colour information is considered in the YCbCr colour space. Here, it has been shown that it makes sense to first reconstruct the Y channel, and to inpaint the two chroma channels Cb and Cr afterwards by relying on the luma channel Y. More precisely, the diffusion tensor is adapted such the inpainting is steered by the luma channel. This helps to preserve the edges from the luma channel also in the chroma channels. For our analysis of the inpainting problem and the optimisation strategies, we mainly focus on grey valued images first for the sake of simplicity.

2.3 Discrete Inpainting Formulation

Digital images reveal a discretisation on an equispaced rectangular grid. Thus, it is natural to formulate the inpainting processes in the discrete setting. To this end, we consider a regular two-dimensional grid $\Gamma = \{0, \ldots, M-1\} \times \{0, \ldots, N-1\}$ with grid size h and the number $|\Gamma| = MN$ of grid locations. A discrete image $\boldsymbol{f} \in \mathbb{R}^{M \times N}$ consists of the values $f_{i,j}$ at the grid points $(i,j) \in \Gamma$ which typically range from 0 to 255. Further, let $K \subset \Gamma$ be the grid points where the image information of $\boldsymbol{f}$ is given, i.e. K denotes the mask points where the Dirichlet boundary conditions are assumed. Often, a binary mask image $\boldsymbol{c} \in \mathbb{R}^{M \times N}$ is utilised to denote known and unknown pixels:

$$c_{i,j} := \begin{cases} 1 & \text{if } (i,j) \in K, \\ 0 & \text{if } (i,j) \in \Gamma \backslash K. \end{cases} \tag{2.9}$$

With this notation, the inpainting solution $\boldsymbol{u} \in \mathbb{R}^{M \times N}$ can be described as the solution of the discrete problem of the form

$$(\boldsymbol{L}\boldsymbol{u})_{i,j} = 0 \qquad \text{for } (i,j) \in \Gamma \backslash K, \tag{2.10}$$

$$u_{i,j} = f_{i,j} \qquad \text{for } (i,j) \in K. \tag{2.11}$$

We define $\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})$ as the function that yields the inpainting solution for a given mask and prescribed grey values. The inpainting solution $\boldsymbol{u}$ highly depends on the discrete symmetric inpainting operator $\boldsymbol{L} : \mathbb{R}^{M \times N} \to \mathbb{R}^{M \times N}$ which yields a 2D image after applying it to a 2D image. As in the continuous setting, a natural choice for $\boldsymbol{L}$ is the discrete Laplace operator $\boldsymbol{A}_N$ fulfilling homogeneous Neumann boundary conditions at the image boundaries. Its stencil notation for the inner grid points reads:

$$\frac{1}{h^2}\begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}\ .$$

It is obtained by means of standard finite differences [139]. The homogeneous Neumann boundary conditions are incorporated by mirroring the image at the boundaries and by using the above stencil also for the boundary grid points. The Laplacian leads to homogeneous diffusion inpainting respectively harmonic inpainting. Similar to the Laplacian, the biharmonic operator $\boldsymbol{B}_N := -\boldsymbol{A}_N^2$ leads to biharmonic inpainting. Of course, one can even consider higher powers of the Laplacian such as the triharmonic operator $\boldsymbol{L} = \boldsymbol{A}_N^3$ or the quadharmonic operator $\boldsymbol{L} = -\boldsymbol{A}_N^4$. Similarly, the EED inpainting operator can be discretised by standard means of finite differences. For more details, see [206].

Instead of homogeneous Neumann boundary conditions, one can impose different conditions at the image boundaries, too. As an example, periodic boundary conditions are sometimes desired. Even though we will not consider inpainting with periodic boundary conditions directly, we will encounter that these boundary conditions are very useful when it comes to improved strategies to find Green's functions. Analogous to the Neumann case, we denote the Laplacian operator incorporating periodic boundary conditions by $\boldsymbol{A}_P$, and the biharmonic counterpart by $\boldsymbol{B}_P$. Also here, is important to mention that all operators $\boldsymbol{A}_N$, $\boldsymbol{B}_N$, $\boldsymbol{A}_P$ and $\boldsymbol{B}_P$ are linear self-adjoint differential operators.

In the literature, it is also common to reorder the 2D images into 1D vectors by collecting the image entries row by row. In this notation, the image values can be addressed by u_k at a pixel index k. Accordingly, the discrete operators are adapted and written in a matrix form. It is straightforward to convert one representation into another, which is why similar notations are utilised within this thesis. From the context and the number of indices, it will be clear whether a 2D or a 1D representation is meant. As a result, one obtains expressions in matrix-vector-notations which makes it possible to employ standard theory from linear algebra. For instance, the discrete Laplacian with homogeneous Neumann boundary conditions leads to a matrix $\boldsymbol{A}_N$ of size $|\Gamma| \times |\Gamma|$ with entries

$$a_{k,\ell} = \begin{cases} \frac{1}{h_d^2} & \text{if } \ell \in \mathcal{N}_\ell, \\ -\sum\limits_{d\in\{x,y\}} \sum\limits_{\ell\in\mathcal{N}_d(k)} \frac{1}{h_d^2} & \text{if } \ell = k, \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

Here, $\mathcal{N}_d(k)$ denotes the neighbours of pixel k in direction d, and h_x, h_y denote the individual grid sizes. This alternative notation allows to write the discrete inpainting problem in (2.10) and (2.11) in an elegant way:

Let $\boldsymbol{C} := \operatorname{diag}(\boldsymbol{c}) \in \mathbb{R}^{|\Gamma|\times|\Gamma|}$ be a matrix by writing all values of the mask image $\boldsymbol{c}$ as diagonal entries, and $\boldsymbol{I} \in \mathbb{R}^{|\Gamma|\times|\Gamma|}$ the identity matrix. Then, we can express the inpainting problem as finding the solution $\boldsymbol{u} \in \mathbb{R}^{|\Gamma|}$ of the following linear system of equations.

$$\boldsymbol{C}(\boldsymbol{u}-\boldsymbol{f})-(\boldsymbol{I}-\boldsymbol{C})\boldsymbol{L}\boldsymbol{u}=\boldsymbol{0}. \tag{2.13}$$

Obviously, this equation enforces the interpolation condition at all mask locations, while the smoothness term is active at all other locations. Due to the binary nature of the mask, only one of the two assumptions is active. Reordering Equation (2.13) yields:

$$\underbrace{(\boldsymbol{C}-(\boldsymbol{I}-\boldsymbol{C})\boldsymbol{L})}_{=:\boldsymbol{M}}\boldsymbol{u}=\boldsymbol{C}\boldsymbol{f}. \tag{2.14}$$

The matrix $\boldsymbol{M}$ is called the inpainting matrix. Note that for the nonlinear EED inpainting, $\boldsymbol{L}=\boldsymbol{L}(\boldsymbol{u})$ and consequently $\boldsymbol{M}=\boldsymbol{M}(\boldsymbol{u})$ depends on the unknown image $\boldsymbol{u}$. Knowing the inverse of $\boldsymbol{M}$ gives direct access to the inpainting solution as $\boldsymbol{u}=\boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{f}$. Unfortunately, the inpainting matrix is not as easy to invert, as we will see in Section 2.5. In practice, the inpainting solution is typically found by employing iterative solvers such as a fast explicit diffusion (FED) scheme [87] or bidirectional multigrid approaches [125]. In the present thesis we will see that it is also possible to obtain the solution directly in a non-iterative way with the help of discrete Green's functions.

2.4 Examples

Before the inpainting problem is analysed in more detail, let us consider some examples. For the experiment, we consider the standard test image *trui* which is shown in Figure 2.2. The common task of the inpainting is to reconstruct an image which approximates the original image as good as possible. It is clear that the inpainting depends on several factors such as the location of mask points, the prescribed tonal values at those locations as well as the inpainting operator. For a prescribed number of mask points, the typical goal is to find the best possible combination of those ingredients. For the moment, however, we only want to get an intuition of the influence of the mask and the inpainting operator on the inpainting result.

We consider harmonic and biharmonic inpainting, and compute the inpainting for fixed masks of varying densities and different configurations. In all cases, the original grey values of the image *trui* are prescribed at the mask locations. First of all, we consider masks with randomly sampled points, and different densities of 1%, 2%, 4% and 16%. These masks are depicted in Figure 2.3 along with the corresponding inpainting results. Besides the random masks, we also consider optimised masks for harmonic inpainting which lead to better reconstructions in general. They are obtained with the continuous theory of Belhachmi *et al.* [23], and contain the same mask densities as the random masks. More details on how to obtain the masks will

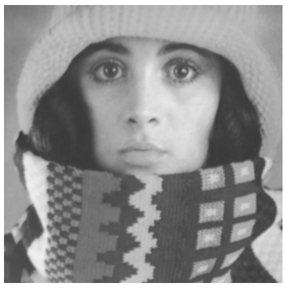

Figure 2.2. Original image *trui* of size 256 × 256.

be discussed in Chapter 5. Again, the masks along with the corresponding inpainting results for different densities are shown in Figure 2.4.

To judge the quality of a given reconstruction it has to be compared to the original image. There are several possibilities to do so. A qualitative comparison can be done by a visual comparison of the reconstructed image to the original image. Alternatively, one can use a quantitative error measure, as for example the commonly used *mean squared error* (MSE). For grey-valued images, it is computed between the original image $\boldsymbol{f}$ and the reconstruction $\boldsymbol{u}$ as follows:

$$\mathrm{MSE}(\boldsymbol{f}, \boldsymbol{u}) := \frac{1}{|\Gamma|} \sum_{(i,j)\in\Gamma} (f_{i,j} - u_{i,j})^2. \tag{2.15}$$

An extension to colour images is straightforward by computing the mean over all channels. A smaller MSE means less differences between the two images and consequently indicates a better reconstruction. Alternatively, the *peak-signal-to-noise ratio* (PSNR) is frequently used as error measure, and is computed based on the MSE:

$$\mathrm{PSNR}(\boldsymbol{f}, \boldsymbol{u}) := 10 \log_{10} \left(\frac{255^2}{\mathrm{MSE}(\boldsymbol{f}, \boldsymbol{u})} \right), \tag{2.16}$$

assuming that the maximum grey-value is 255. The PSNR is given in dB, and a larger value means a better quality of the reconstruction. Since the MSE and the PSNR lead to the same rating order, it suffices to only consider one error measure. The results show that the error measure is a very good indicator for the visual quality.

It is clear that a higher mask density also leads to better reconstructions due to the increased availability of information. This is both visible from the qualitative as well as from the quantitative judgement. Moreover, one can state that the optimised masks give much better results than the random masks except for the very sparse mask. The reconstructions with the random masks are much more blurry. Another observation from the random masks is that biharmonic inpainting gives better reconstructions than its harmonic counterpart. It is clear that this is not the case for the optimised masks, as they are tuned for homogeneous diffusion inpainting.

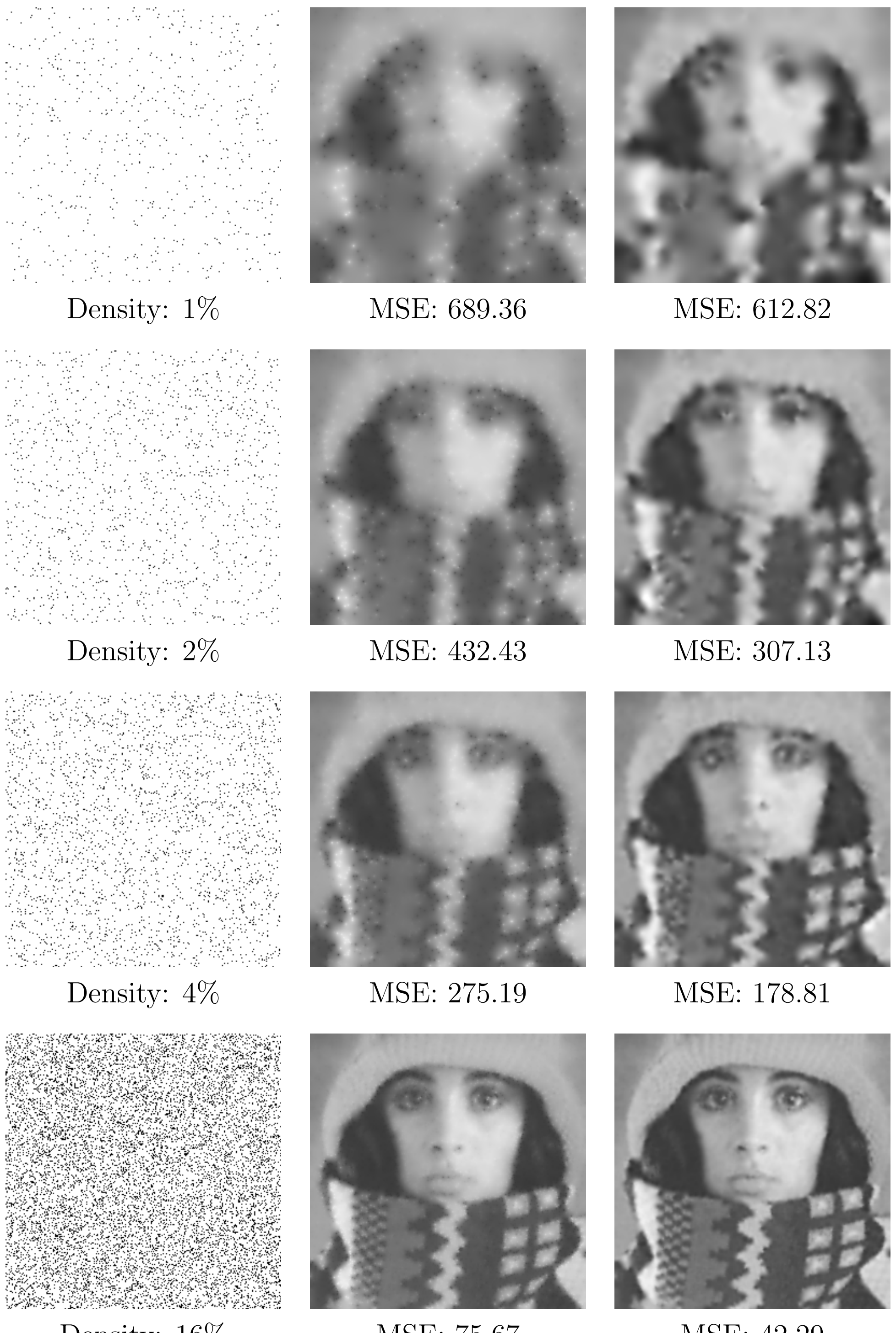

Figure 2.3. Examples of random masks with different densities. **Left column:** Given masks. **Centre column:** Harmonic inpainting results. **Right column:** Biharmonic inpainting results.

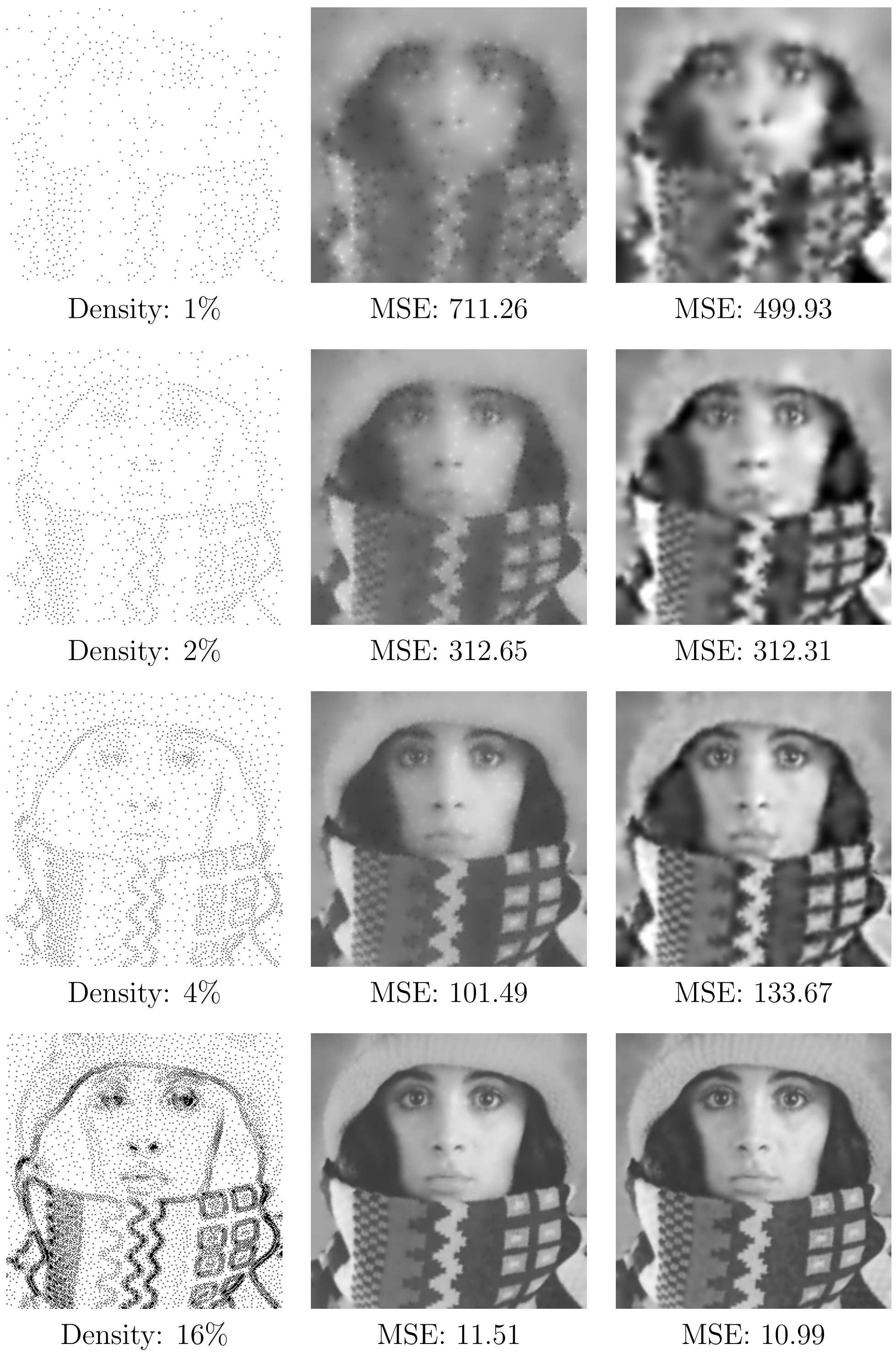

Figure 2.4. Examples of optimised masks for the image *trui* with different densities obtained by electrostatic halftoning, see Chapter 5. **Left column:** Given masks. **Centre column:** Harmonic inpainting results. **Right column:** Biharmonic inpainting results.

In total, the main message of this experiment is that it pays off to optimise the individual ingredients of the inpainting process. In particular, the experiments indicate an improved reconstruction quality with an optimised mask. It seems that the biharmonic inpainting leads to better results. However, the final statement on the reconstruction ability of the operators requires a fair comparison with optimised mask and tonal values for the respective operators.

2.5 Analysis of the Discrete Inpainting Problem

Both the complexity of finding the inpainting solution as well as the properties of the inpainting mainly depend on the inpainting matrix $\boldsymbol{M}$. In the following, these aspects will be investigated further by also gaining a deeper understanding of the inpainting matrix. Thereby, we focus on inpainting with linear inpainting operators along with homogeneous Neumann boundary conditions, as they will be within the main focus throughout this thesis. Moreover, we assume the mask to be non-empty, i.e. there should be at least one mask point. This obviously makes sense and leads to a well-posed problem. Note that the general properties have already been derived in previous publications. Nevertheless, we recapitulate them because they are important for later investigations. Additionally, given these existing results, it is easier to integrate the novel findings into a global context. After the general properties, we analyse the inpainting matrix further by considering its condition number as well as an extension to non-binary masks.

2.5.1 General Properties

The inpainting solution reveals some interesting properties. Most importantly, it has been shown that the solution of the discrete inpainting problem with homogeneous diffusion inpainting exists and is unique [125]. Moreover, it could be proven that the inpainting solution fulfils the maximum-minimum principle, i.e. all values of the inpainting solution lie between the minimum and maximum of the prescribed values [125]:

$$\min_{(k,l)\in K} f_{k,l} \leq u_{i,j} \leq \min_{(k,l)\in K} f_{k,l} \qquad \text{for all } (i,j) \in \Gamma. \tag{2.17}$$

When it comes to biharmonic inpainting, the maximum-minimum principle is not guaranteed anymore. Over- and undershoots typically lead to values in the reconstruction that exceed the range of the prescribed values by far [41]. Further properties of the inpainting are elaborated later in this thesis. To this end, discrete Green's functions will play a key role for obtaining a better understanding.

2.5.2 Condition Numbers

Given a general linear system of equations of the form $\boldsymbol{Ax} = \boldsymbol{b}$, the condition number is a standard measure to rate the complexity of finding a solution [165, 134]. Intuitively, a larger condition number means that changes in $\boldsymbol{b}$ have a higher impact on the solution $\boldsymbol{x}$, and vice versa. Thus, the higher the condition number is, the less accurate an approximation of the solution will be. In other words, if the condition number is small, the matrix $\boldsymbol{A}$ is well conditioned and so computing its inverse is possible with high accuracy. In contrast, if the condition number becomes larger, the system matrix becomes more and more ill-conditioned and harder to invert. When it comes to iterative numerical solvers for linear system of equations, the condition number gives a good hint about the convergence rate of the scheme. More formally, the condition number $\kappa(\boldsymbol{A})$ in the ℓ_2 norm is specified as

$$\begin{aligned}\kappa(\boldsymbol{A}) &= \|\boldsymbol{A}\|_2 \cdot \|\boldsymbol{A}^{-1}\|_2 \\ &= \frac{\sigma_{max}(\boldsymbol{A})}{\sigma_{min}(\boldsymbol{A})}. \end{aligned} \tag{2.18}$$

The matrix norm $\|\boldsymbol{A}^{-1}\|_2$ might be hard to estimate due to the need to invert the matrix. Instead, it is easier to compute the condition number based on the maximal and minimal singular values σ_{max} and σ_{min} of $\boldsymbol{A}$. They in turn are given by considering the maximum and minimum eigenvalues λ_{max} and λ_{min} of $\boldsymbol{A}^\top\boldsymbol{A}$ (cf. [136]):

$$\sigma_{max} = \sqrt{\lambda_{max}}\,, \qquad \sigma_{min} = \sqrt{\lambda_{min}}\,. \tag{2.19}$$

Note that λ_{max} is the spectral radius of the matrix $\boldsymbol{A}^\top\boldsymbol{A}$. One possibility to estimate this spectral radius is to use Gershgorin's circle theorem. However, this may give a too inaccurate estimate. Instead, one can employ the power method to determine the maximum eigenvalue of $\boldsymbol{A}^\top\boldsymbol{A}$. For more details, please refer to [11]. Moreover, the minimum eigenvalue λ_{min} of $\boldsymbol{A}^\top\boldsymbol{A}$ can be found with the help of the power method, too. To this end, we define a new matrix $\boldsymbol{B} := \boldsymbol{A}^\top\boldsymbol{A} - \lambda_{max}\boldsymbol{I}$, and obtain the eigenvalue μ_{max} of $\boldsymbol{B}$ with the largest magnitude by the power method. Then, Theorem 2.1 allows get direct access to λ_{min}:

Theorem 2.1. *The minimum eigenvalue of $\boldsymbol{A}^\top\boldsymbol{A}$ is given by $\lambda_{min} = \mu_{max} + \lambda_{max}$.*

Proof. Let λ be an eigenvalue of $\boldsymbol{A}^\top\boldsymbol{A}$ with corresponding eigenvector $\boldsymbol{v}$. Then, we compute

$$\begin{aligned}\boldsymbol{Bv} &= (\boldsymbol{A}^\top\boldsymbol{A} - \lambda_{max}\boldsymbol{I})\boldsymbol{v} \\ &= \boldsymbol{A}^\top\boldsymbol{Av} - \lambda_{max}\,\boldsymbol{v} \\ &= \lambda\boldsymbol{v} - \lambda_{max}\,\boldsymbol{v} \\ &= (\lambda - \lambda_{max})\boldsymbol{v} \end{aligned} \tag{2.20}$$

This shows that $\mu := \lambda - \lambda_{max}$ is an eigenvalue of $\boldsymbol{B}$. Since the matrix $\boldsymbol{A}^\top \boldsymbol{A}$ is positive semi-definite, its eigenvalues are all larger or equal zero, and range from λ_{min} to λ_{max}. Consequently, the eigenvalues of $\boldsymbol{B}$ range from $\lambda_{min} - \lambda_{max}$ to zero, with $\mu_{max} = \lambda_{min} - \lambda_{max}$ being the eigenvalue of largest magnitude. Thus, the power method can be employed to find μ_{max}, which in turn yields $\lambda_{min} = \mu_{max} + \lambda_{max}$. $\square$

In total, we can evaluate the condition number of the the inpainting matrix $\boldsymbol{M}$ without the need of inverting it. Note that the condition number only depends on the mask and the inpainting operator. In an experiment, the condition number is computed for random masks as well as optimised masks obtained by the theory of Belhachmi *et al.* [23], with densities ranging from 0.01% to 16%. For some densities the masks have already been shown in Figures 2.3 and 2.4. An overview of the resulting condition numbers of the inpainting operators for harmonic and biharmonic inpainting is presented in Table 2.1.

Table 2.1. Condition numbers of inpainting matrix $\boldsymbol{M}$ for harmonic and biharmonic inpainting and for different masks with varying densities.

	random mask		optimised mask	
mask density	harmonic inpainting	biharmonic inpainting	harmonic inpainting	biharmonic inpainting
0.01 %	142999	1878688037	126149	1331213279
0.50 %	3258	1009367	8678	11034983
1.00 %	1741	249021	4133	1979904
2.00 %	895	53707	2163	574572
4.00 %	578	31141	1049	79383
8.00 %	286	8618	625	21656
16.00 %	147	2169	389	6416

As we can see, the condition numbers increase tremendously the sparser the mask becomes: While the inpainting problem is easy to handle for masks with a density around 16%, the problem becomes very complex for masks containing less than 1% of prescribed points. Additionally, it is obvious that the biharmonic inpainting problem is much worse than the corresponding harmonic inpainting problem, which is not surprising as the power of the differential operator is squared. One has to keep this in mind when talking about which operator is better suited for inpainting and compression tasks. Although the biharmonic operator might lead to qualitatively better results, the harmonic one is probably easier to handle by numerical solvers. This could in turn lead to reduced runtimes.

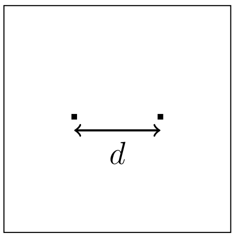

Figure 2.5. Sketch of mask with only two points with varying distance d between the two mask points.

Another observation is that the condition number for the optimised masks is usually higher than for the corresponding random mask with the same density. The reason for this fact is that the mask points do not capture all regions of the image anymore. The resulting holes make it necessary for the inpainting to diffuse the known information over longer distances. Although theoretically each mask point has a global influence on the inpainting result, the impact is limited by the surrounding mask points. A detailed analysis on this topic will be done in Section 4.2.3. Consequently, whenever there is a larger gap between neighbouring mask points, the condition number increases. Obviously, filling in such large unknown areas is rather expensive. This also explains the success of bidirectional multigrid methods: On coarse scales, they reduce the distance between the mask points, which makes solving the problem at those levels much easier. Although the difference of the condition number for different mask configurations is by far not as severe as the complexity difference between the harmonic and biharmonic inpainting, one also has to keep this in mind when solving the inpainting problem. Besides an increased runtime, stopping criteria potentially have to be adapted accordingly.

From the analysis above, one can jump to the conclusion that the worst case condition number is obtained with a mask containing only two points. We do not want to discuss the case with only one mask point as the solution is trivially given by a constant image with the prescribed grey-value everywhere. Instead, let us consider a family of masks as sketched in Figure 2.5. For an image domain of size 256×256, the two mask points are vertically centred and have a varying distance d between. For these masks, the corresponding condition numbers of the inpainting operator with both the harmonic and biharmonic inpainting operator are presented in Table 2.2.

As expected, the condition numbers in this scenario are very large, and also larger than all condition numbers for the random or optimised masks before. Normally, from the reasoning above, one would expect that the largest condition number occurs with the largest tested distance of 240. However, the homogeneous Neumann boundary conditions also play a major role in this extreme example. A larger d leads to the mask points being closer to the boundary. Here, the mask points influence themselves. Thus, for the bipole mask with $d = 0$, the distance between the points is minimal, but the distance to the mirrored points is maximised, and so is the condition number.

Table 2.2. Condition numbers of inpainting matrices for two point masks with varying distance.

distance d [pixel]	harmonic inpainting	biharmonic inpainting
0	448238	3880942165
40	295447	3594948006
80	269002	3545260973
120	260355	3527687198
160	262638	3531151910
200	280965	3550184839
240	331336	3579171982

2.6 Extension to Non-Binary Masks

So far, the inpainting mask has always assumed to be binary. However, as it is proposed in [92], it is also possible to use non-binary mask values. This is possible by reconsidering the inpainting formulation in (2.13):

$$\boldsymbol{C}(\boldsymbol{u}-\boldsymbol{f})+(\boldsymbol{I}-\boldsymbol{C})(-\boldsymbol{L}\boldsymbol{u})=\boldsymbol{0}. \tag{2.21}$$

By choosing the mask values to be one or zero, one can select between the two assumptions: Either the Dirichlet boundary condition is imposed, or the smoothness of the inpainting solution is enforced. The left hand side of the equation can also be understood as a convex combination of these two assumptions. By choosing continuous values of $\boldsymbol{c}$ in $[0,1]$, a mixture between the two extremes can be obtained. For example, this makes sense if one does not fully trust the data, and also wants to smooth the inpainting result at the mask points. This comes down to a forward diffusion at those mask points. Besides a pure interpolation between the two assumptions, it is also possible to extrapolate by selection mask values outside the range $[0,1]$. Whenever the mask value is larger than one, the interpolation condition is more than enforced. In other words, one obtains a backward diffusion at those locations.

In [92], this idea of using continuous mask values is employed to find an optimal (binary) mask for the inpainting problem with the Laplacian $\boldsymbol{A}_N$ as inpainting operator. To this end, Equation (2.21) is used as a side constraint of an energy functional. The final mask is found with the help of a primal-dual approach [156]. For more details, please see [92]. However, the important question is: Given a fixed continuous mask $\boldsymbol{c}$, is it always possible to obtain a corresponding inpainting solution by solving the inpainting equation (2.21)? In the original paper, this is required in each iteration step to obtain the up-to-date inpainting solution for an fixed intermediate mask (cf. Algorithm 2 in [92]), and also in the end to obtain a final inpainting with the continuous mask. Unfortunately, the answer to the question is no. In fact, the resulting inpainting matrix $\boldsymbol{M} = \boldsymbol{C} - (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{A}_N$ can be singular for certain masks. This is shown in Theorem 2.2.

Theorem 2.2 (Singular Inpainting matrix for Continuous Mask Values)**.** *The matrix $\boldsymbol{M}$ is not invertible for arbitrary $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$, even for $c \neq 0$.*

Proof. The case with $\boldsymbol{c} = \boldsymbol{0}$ is trivial as the inpainting operator $\boldsymbol{L}$ has a zero eigenvalue with a constant image as corresponding eigenvector. Let us therefore consider the case $\boldsymbol{c} \neq \boldsymbol{0}$. As a counter example, consider an image with 6 pixels and a grid size of $h = 1$:

u_1	u_2	u_3
u_4	u_5	u_6

Here, the matrix for the discrete Laplace operator $\boldsymbol{A}_N$ has the following form:

$$\boldsymbol{A}_N = \begin{pmatrix} -2 & 1 & 0 & 1 & 0 & 0 \\ 1 & -3 & 1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 1 & 0 \\ 0 & 1 & 0 & 1 & -3 & 1 \\ 0 & 0 & 1 & 0 & 1 & -2 \end{pmatrix}. \tag{2.22}$$

Now, we define a mask with the following values:

$$\boldsymbol{C} = \operatorname{diag}\left((1.5, 1, 1, 0, 4, 5)^\top\right). \tag{2.23}$$

The matrix $\boldsymbol{M}$ has as the eigenvalue 0, since the vector $\boldsymbol{u} = (1, 0, 0, -1, -3, -4)^\top$ is in the kernel of $\boldsymbol{M}$, i.e., $\boldsymbol{M}\boldsymbol{u} = \boldsymbol{0}$ for a non-trivial $\boldsymbol{u}$. It follows that $\boldsymbol{M}$ is not singular. $\square$

This shows that it is not possible to obtain an inpainting for arbitrary continuous mask values. In practice, it is not unlikely that this happens during the process of finding an optimised mask with the algorithm in [92]. One remedy would be to compute the Moore-Penrose pseudo-inverse of $\boldsymbol{M}$ instead the real inverse. However, it is not clear if this leads to a desired solution. Luckily, there is another remedy to this issue: It is possible to avoid a singular inpainting matrix for continuous mask values by restricting the possible range. Theorem 2.3 gives a proof for common images with a grid size of $h = 1$. The corresponding considerations for general grid sizes can be found in [91], Section 2.2.

Theorem 2.3. *For grid sizes of $h = 1$, the matrix $\boldsymbol{M} = \boldsymbol{C} - (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{A}_N$ is invertible with $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$, if $0 \leq c_i \leq \frac{8}{7}$ for all mask values c_i and $\boldsymbol{c}$ is not equal to the zero everywhere.*

Proof. It is sufficient to show that 0 is not an eigenvalue in those cases. We first consider a mask location i which is not at the boundary. Considering the Gerschgorin disks of $\boldsymbol{M}$, we know that an eigenvalue λ lies within a disk that is given by

$$|\lambda - (c_i + 4(1 - c_i))| \leq 4|1 - c_i|. \tag{2.24}$$

In order for 0 not lying in one of the disks and thus not being an eigenvalue, the following has to hold true.

$$|c_i + 4(1 - c_i)| > 4|1 - c_i|. \tag{2.25}$$

We can distinguish the following three cases.

Case 1: $c_i + 4(1 - c_i) < 0 \Leftrightarrow c_i > \frac{4}{3}$. It follows $c_i > 1$ and thus:

$$-(4 - 3c_i) > -4(1 - c_i) \qquad \Leftrightarrow \qquad c_i < 0.$$

This is a contradiction and therefore, this case cannot occur.

Case 2: $c_i + 4(1 - c_i) > 0 \Leftrightarrow c_i < \frac{4}{3}$. This leads to the following cases

$$\begin{aligned}
c_i > 1: &\qquad 4 - 3\,c_i > -4(1 - c_i) &\qquad \Leftrightarrow &\qquad c_i < \frac{8}{7},\\
c_i < 1: &\qquad 4 - 3\,c_i > 4(1 - c_i) &\qquad \Leftrightarrow &\qquad c_i > 0,\\
c_i = 1: &\qquad 4 - 3\,c_i > 4(1 - c_i) &\qquad \Leftrightarrow &\qquad 1 > 0.
\end{aligned}$$

Case 3: $c_i + 4(1 - c_i) = 0 \Leftrightarrow c_i = \frac{4}{3}$. It follows $0 > \frac{4}{3}$, which again is a contradiction.

It follows that $0 < c_i < \frac{8}{7}$ must hold. Analogously, the same computations can be done for boundary pixels, leading to the same result. In order to include $c_i = 0$ and $c_i = \frac{8}{7}$ as well, one can argue in the same way as in Mainberger *et al.* [125] in the proof of Theorem 1: Since $\boldsymbol{M}$ is block irreducible, the theorem of Feingold and Varga [72] implies that an eigenvalue lying on the boundary of the union of all Gerschgorin circles must lie in all individual Gerschgorin circles. As we assume that there is at least one mask point, i.e. there exists one i with $c_i > 0$, there is at least one Gerschgorin circle that does not contain zero. Similarly, it can be shown that there exists one circle that does not contain $\frac{8}{7}$. Hence, these two border cases can be included and the $\boldsymbol{M}$ is still invertible. □

As a consequence of this theorem, one has to adapt the algorithm in [92] to enforce the mask values to lie within the range of $[0, \frac{8}{7}]$. Otherwise, the inpainting result may not be well-defined. This can be achieved by a back projection of the mask values into this range after each update. This means that all values are mapped to the closest value within the range. Interestingly, a binary mask is just a special case where all mask values are within the admissible range. Thus, the problem does not occur after thresholding, and confirms the existence result of the inpainting from [125].

Chapter 3

Discrete Green's Functions for PDE-based Inpainting

3.1 Motivation

In the present Section, we want to gain more theoretical insights on inpainting methods with linear self-adjoint differential operators such as the harmonic or biharmonic operator. In particular, we analyse their relation to a very popular idea in modern signal and image analysis, namely sparsity. To this end, we make use of the concept of discrete Green's functions [24, 52]. Green's functions are mainly known from the continuous theory of partial differential equations (PDEs) as a tool to describe the solution of boundary value problems [135]. Most publications on Green's functions focus on continuous differential operators. In [178], continuous Green's functions have already been used to perform image inpainting. However, the authors did not pay any attention to the boundary conditions.

Digital images, on the other hand, reveal a natural discretisation on a regular grid. Moreover, they are given on a rectangular image domain, and it is fairly common to extend image processing operators at the boundaries by mirroring. This motivates us to investigate discrete Green's functions for linear differential operators on a rectangular image domain with homogeneous Neumann boundary conditions. Moreover, we will give an interpretation of the obtained discrete Green's functions in terms of linear algebra. More precisely, we will elaborate the connection to the Moore–Penrose inverse of the discretised differential operator.

Besides gaining novel theoretical insights with the help of the discrete Green's functions, we will also investigate potential algorithmic benefits. As we will see, a fast way to generate the discrete Green's functions and to compute inner products is indispensable to obtain efficient alternative algorithms for solving the inpainting problem.

The structure of the present chapter is as follows. First, we consider the eigenvalues and eigenvectors of the inpainting operators in Section 3.2. They are crucial for later tasks. In the subsequent Section 3.3 we introduce the theory of discrete Green's functions and propose efficient algorithms. Section 3.4 discusses how the discrete Green's functions allow to find the solution of the inpainting problem. Numerical advantages of our Green's function framework are discussed in Section 3.5. This chapter is concluded with a summary in Section 3.6. Note that most of the presented contents are published in [3].

3.2 Eigenvalues and Eigenvectors of the Inpainting Operators

For our later analysis it is useful to represent the discrete inpainting operators, the discrete Laplacian $\boldsymbol{A}_N$ and the biharmonic operator $\boldsymbol{B}_N$, in terms of their eigenvalues and eigenvectors. The following theorem provides the required information. It extends 1D results that can be found for example in [165, 186] to the two-dimensional setting.

Theorem 3.1 (Eigenvalues and Eigenvectors of the Discrete Operators with Homogeneous Neumann Boundary Conditions)**.** *The orthonormal set of eigenvectors of $\boldsymbol{A}_N$ as well as of $\boldsymbol{B}_N$ is given for all $(m,n) \in \Gamma$ by*

$$(\boldsymbol{v}^N_{m,n})_{i,j} = \begin{cases} \sqrt{\frac{1}{MN}} & \text{if } m = n = 0, \\ \sqrt{\frac{2}{MN}} \cos\left(\frac{m\pi\left(i+\frac{1}{2}\right)}{M}\right) \cos\left(\frac{n\pi\left(j+\frac{1}{2}\right)}{N}\right) & \text{if either } m = 0 \text{ or } n = 0, \\ \sqrt{\frac{4}{MN}} \cos\left(\frac{m\pi\left(i+\frac{1}{2}\right)}{M}\right) \cos\left(\frac{n\pi\left(j+\frac{1}{2}\right)}{N}\right) & \text{if } m > 0 \text{ and } n > 0. \end{cases} \tag{3.1}$$

The corresponding eigenvalues for the discrete Laplace operator $\boldsymbol{A}_N$ are

$$\lambda^{\boldsymbol{A}_N}_{m,n} = -\frac{4}{h^2}\left(\sin^2\left(\frac{m\pi}{2M}\right) + \sin^2\left(\frac{n\pi}{2N}\right)\right). \tag{3.2}$$

The eigenvalues of the discrete biharmonic operator $\boldsymbol{B}_N$ read as

$$\lambda^{\boldsymbol{B}_N}_{m,n} = -\left(\lambda^{\boldsymbol{A}_N}_{m,n}\right)^2. \tag{3.3}$$

Proof. While this eigenstructure may not appear obvious, proving its correctness is fairly straightforward: One has to check that $\boldsymbol{A}_N \boldsymbol{v}_{m,n} = \lambda^{\boldsymbol{A}_N}_{m,n} \boldsymbol{v}^N_{m,n}$ and $\boldsymbol{B}_N \boldsymbol{v}_{m,n} = \lambda^{\boldsymbol{B}_N}_{m,n} \boldsymbol{v}^N_{m,n}$ hold true for all $(m,n) \in \Gamma$ and that the discrete homogeneous Neumann boundary conditions are fulfilled. Additionally, one has to show the orthonormality of the set of eigenvectors. □

Note that both operators are singular, since the eigenvalues $\lambda_{0,0}^{\boldsymbol{A}_N}$ and $\lambda_{0,0}^{\boldsymbol{B}_N}$ vanish. This will complicate some of our discussions on discrete Green's functions in the next section. An important observation is that these eigenvectors exactly represent the basis vectors of the discrete cosine transform, more precisely of type II (cf. [159, 165]). They are widely used, also in JPEG for example. We will come back to this fact in Section 3.3.4 where an efficient way to compute inner products with discrete Green's functions is presented.

Besides the Neumann boundary conditions, we also want to have a look at the eigenvectors and eigenvalues of the operators with periodic boundary conditions. They can be written for the 2D setting in the following form [179]:

Theorem 3.2 (Eigenvalues and Eigenvectors of the Discrete Operators with Periodic Boundary Conditions)**.** *The orthonormal set of eigenvectors of $\boldsymbol{A}_P$ as well as of $\boldsymbol{B}_P$ is given for all $(m,n) \in \Gamma$ by*

$$(\boldsymbol{v}_{m,n}^P)_{i,j} = (\boldsymbol{x}_m)_i \cdot (\boldsymbol{y}_n)_j \tag{3.4}$$

with the vectors $\boldsymbol{x}_m$ and $\boldsymbol{y}_n$ defined as

$$(\boldsymbol{x}_m)_i = \begin{cases} \sqrt{\frac{1}{M}} & \textit{if } m = 0, \\ \sqrt{\frac{2}{M}} \cos\left(\frac{m\pi\left(i+\frac{1}{2}\right)}{M}\right) & \textit{if } m \textit{ even and } m > 0, \\ \sqrt{\frac{2}{M}} \sin\left(\frac{(m+1)\pi\left(i+\frac{1}{2}\right)}{M}\right) & \textit{if } m \textit{ odd, } m > 0 \textit{ and } m+1 < M, \\ \sqrt{\frac{1}{M}} \sin\left(\frac{(m+1)\pi\left(i+\frac{1}{2}\right)}{M}\right) & \textit{if } m \textit{ odd, } m > 0 \textit{ and } m+1 = M, \end{cases} \tag{3.5}$$

and

$$(\boldsymbol{y}_n)_j = \begin{cases} \sqrt{\frac{1}{N}} & \textit{if } n = 0, \\ \sqrt{\frac{2}{N}} \cos\left(\frac{n\pi\left(j+\frac{1}{2}\right)}{N}\right) & \textit{if } n \textit{ even and } n > 0, \\ \sqrt{\frac{2}{N}} \sin\left(\frac{(n+1)\pi\left(j+\frac{1}{2}\right)}{N}\right) & \textit{if } n \textit{ odd, } n > 0 \textit{ and } n+1 < N, \\ \sqrt{\frac{1}{N}} \sin\left(\frac{(n+1)\pi\left(j+\frac{1}{2}\right)}{N}\right) & \textit{if } n \textit{ odd, } n > 0 \textit{ and } n+1 = N. \end{cases} \tag{3.6}$$

The corresponding eigenvalues for the discrete Laplace operator $\boldsymbol{A}_P$ are

$$\lambda_{m,n}^{\boldsymbol{A}_P} = \begin{cases} -\frac{4}{h^2}\left(\sin^2\left(\frac{m\pi}{2M}\right) + \sin^2\left(\frac{n\pi}{2N}\right)\right) & \textit{if } m, n \textit{ even}, \\ -\frac{4}{h^2}\left(\sin^2\left(\frac{m\pi}{2M}\right) + \sin^2\left(\frac{(n+1)\pi}{2N}\right)\right) & \textit{if } m \textit{ even}, n \textit{ odd}, \\ -\frac{4}{h^2}\left(\sin^2\left(\frac{(m+1)\pi}{2M}\right) + \sin^2\left(\frac{n\pi}{2N}\right)\right) & \textit{if } m \textit{ odd}, n \textit{ even}, \\ -\frac{4}{h^2}\left(\sin^2\left(\frac{(m+1)\pi}{2M}\right) + \sin^2\left(\frac{(n+1)\pi}{2N}\right)\right) & \textit{if } m, n \textit{ odd}. \end{cases} \tag{3.7}$$

The eigenvalues of the discrete biharmonic operator $\boldsymbol{B}_P$ read as

$$\lambda_{m,n}^{\boldsymbol{B}_P} = -\left(\lambda_{m,n}^{\boldsymbol{A}_P}\right)^2 . \tag{3.8}$$

Proof. The proof is analogous to the previous proof for the operators with homogeneous Neumann boundary conditions. □

Similar to before, this set of eigenvectors can be interpreted as the real and complex parts of the well known discrete Fourier transform [180, 185, 32]. In contrast to these real eigenvectors, one can also choose the eigenvectors in the complex domain. The eigenvalues, however, remain real, as the following theorem shows:

Theorem 3.3 (Eigenvalues and Complex Eigenvectors of the Discrete Operators with Periodic Boundary Conditions)**.** *The orthonormal set of complex eigenvectors of $\boldsymbol{A}_P$ as well as of $\boldsymbol{B}_P$ is given for all $(m,n) \in \Gamma$ by the discrete Fourier basis:*

$$(\boldsymbol{v}_{m,n}^{P})_{i,j} = \sqrt{\frac{1}{MN}} \cdot e^{-\frac{2\iota\pi m i}{M}} \cdot e^{-\frac{2\iota\pi n j}{N}} , \tag{3.9}$$

where ι denotes the imaginary unit. The eigenvalues for the discrete Laplace operator $\boldsymbol{A}_P$ are given as

$$\lambda_{m,n}^{\boldsymbol{A}_P} = -\frac{4}{h^2}\left(\sin^2\left(\frac{m\pi}{M}\right) + \sin^2\left(\frac{n\pi}{N}\right)\right) , \tag{3.10}$$

and for the discrete biharmonic operator $\boldsymbol{B}_P$ as

$$\lambda_{m,n}^{\boldsymbol{B}_P} = -\left(\lambda_{m,n}^{\boldsymbol{A}_P}\right)^2 . \tag{3.11}$$

A proof can for example be found in [86].

3.3 Discrete Green's Functions

In this section, we introduce discrete Green's functions, and discuss important properties of them. This will be very useful for describing, understanding, analysing as well as solving discrete inpainting problems.

3.3.1 Definition and Properties

Let us consider a general discrete problem of the following type:

$$\boldsymbol{L}\boldsymbol{u} = \boldsymbol{a}. \tag{3.12}$$

Thereby, $\boldsymbol{u} \in \mathbb{R}^{M\times N}$ is the unknown image, $\boldsymbol{a} \in \mathbb{R}^{M\times N}$ is a prescribed right hand side, and $\boldsymbol{L} : \mathbb{R}^{M\times N} \to \mathbb{R}^{M\times N}$ a given symmetric self-adjoint discrete linear differential operator incorporating some specific boundary conditions.

Usually, the solution to the discrete problem (3.12) is found by solving a linear system of equations. As we will see, discrete Green's functions represent a very useful tool to obtain direct access to the solution. Besides this property, they are also important to understand and analyse discrete problems as we will see later. Their definition is given as follows:

Definition 3.4 (Discrete Green's Functions)**.** If $\boldsymbol{L}$ is invertible, the discrete Green's function $\boldsymbol{g}_{k,\ell}$ corresponding to a source point $(k,\ell) \in \Gamma$ is defined as the solution of

$$(\boldsymbol{L}\boldsymbol{g}_{k,\ell})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} \qquad \text{for } (i,j) \in \Gamma, \tag{3.13}$$

with the Kronecker delta function

$$(\boldsymbol{\delta}_{k,\ell})_{i,j} = \begin{cases} 1 & \text{if } (i,j) = (k,\ell), \\ 0 & \text{if } (i,j) \neq (k,\ell). \end{cases} \tag{3.14}$$

Otherwise, if $\boldsymbol{v}$ is the eigenvector of $\boldsymbol{L}^\top$ corresponding to a single eigenvalue 0, the infinitely many discrete Green's functions for a point $(k,\ell) \in \Gamma$ are defined as solutions of

$$(\boldsymbol{L}\boldsymbol{g}_{k,\ell})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} - \frac{v_{i,j} \cdot v_{k,\ell}}{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} \qquad \text{for } (i,j) \in \Gamma. \tag{3.15}$$

Here, the Euclidean inner product $\langle \boldsymbol{a}, \boldsymbol{b} \rangle := \sum_{(i,j)\in\Gamma} a_{i,j} b_{i,j}$ between two images $\boldsymbol{a}$ and $\boldsymbol{b}$ is defined in a standard way. As an intuition, the Green's function can be considered as the influence of an impulse at a point (k,ℓ) on the complete image.

Interestingly, the solution of the discrete problem (3.12) is obtained as a simple superposition of these discrete Green's functions (cf. [68]):

Theorem 3.5 (Analytic Solution)**.** *Let $\boldsymbol{g}_{k,\ell}$ denote the discrete Green's functions at positions $(k,\ell) \in \Gamma$. Then, the solution $\boldsymbol{u}$ of the discrete problem (3.12) is given by*

$$\boldsymbol{u} = \sum_{(k,\ell)\in\Gamma} a_{k,\ell}\, \boldsymbol{g}_{k,\ell}. \tag{3.16}$$

Hence, once the Green's functions are known, we are able to write down the solution directly. However, finding them is not always a trivial task as they depend on the domain as well as on the boundary conditions. This aspect is discussed in more detail in Section 3.3.3.

The case distinction in the definition of the discrete Green's functions ensures the existence of the Green's functions. This can be seen with the so called Fredholm alternative, which states under which conditions a discrete problem has a solution (cf. [55]).

Theorem 3.6 (Fredholm Alternative)**.** *If $\boldsymbol{L}$ is invertible, then the solution $\boldsymbol{u}$ of the discrete problem (3.12) exists and is unique. Otherwise, assuming that $\boldsymbol{L}^\top$ possesses the single eigenvalue 0 with the corresponding eigenvector $\boldsymbol{v} \in \mathbb{R}^{M\times N}$, there exist infinitely many solutions if*

$$\langle \boldsymbol{v}, \boldsymbol{a} \rangle = 0, \tag{3.17}$$

and there exists no solution at all if

$$\langle \boldsymbol{v}, \boldsymbol{a} \rangle \neq 0. \tag{3.18}$$

With this theorem it is straightforward to verify the existence of the Green's functions given in Definition 3.4. In particular, note that there are infinitely many Green's functions in case of a singular operator $\boldsymbol{L}$. A nice property of the discrete Green's function is their symmetry, which will become important in later sections:

Theorem 3.7 (Symmetry of Discrete Green's Functions)**.** *Let $(i,j),(k,\ell) \in \Gamma$. Then, the discrete Green's functions fulfill*

$$(\boldsymbol{g}_{k,\ell})_{i,j} = (\boldsymbol{g}_{i,j})_{k,\ell}\,. \tag{3.19}$$

Proof. The symmetry follows directly from the definition of the discrete Green's functions and from the symmetry of the Kronecker delta function. □

3.3.2 Interpretation as Moore–Penrose Inverse

The Fredholm alternative can also be expressed in terms of linear algebra. For this analysis, make use of the alternative matrix vector notation as in Section 2.3. This means, $\boldsymbol{u}$ and $\boldsymbol{a}$ are considered as vectors of length MN, and $\boldsymbol{L}$ becomes a symmetric $(MN \times MN)$-matrix. Then, (3.12) transfers to a linear system $\boldsymbol{L}\,\boldsymbol{u} = \boldsymbol{a}$ of size MN. This system is uniquely solvable, if and only if $\boldsymbol{L}$ is invertible. If $\operatorname{rank}(\boldsymbol{L}) = MN - 1$, then (3.12) possesses either infinitely many solutions if $\operatorname{rank}(\boldsymbol{L}) = \operatorname{rank}(\boldsymbol{L}, \boldsymbol{a})$, or no solution if $\operatorname{rank}(\boldsymbol{L}) < \operatorname{rank}(\boldsymbol{L}, \boldsymbol{a})$.

Assuming that $\boldsymbol{L}$ is invertible, the discrete Green's function defined in (3.13) can be expressed as the solution of

$$\boldsymbol{L}\,\boldsymbol{G} = \boldsymbol{I}, \tag{3.20}$$

where $\boldsymbol{I}$ again denotes the identity matrix of size $MN \times MN$ and $\boldsymbol{G} \in \mathbb{R}^{MN\times MN}$ the matrix that contains the discrete Green's functions $\boldsymbol{g}_{k,\ell}$ as columns.

If $\operatorname{rank}(\boldsymbol{L}) = MN - 1$ then there exist infinitely many Green's functions, and (3.15) leads to:

$$\boldsymbol{L}\,\boldsymbol{G} = \boldsymbol{I} - \frac{1}{\langle \boldsymbol{v}, \boldsymbol{v} \rangle}(\boldsymbol{v})(\boldsymbol{v})^\top, \tag{3.21}$$

with $\boldsymbol{v} \in \mathbb{R}^{MN}$. In the following theorem, a useful additional constraint is introduced that creates a unique solution and allows to relate discrete Green's functions to the Moore–Penrose inverse of their discrete differential operator. The Moore–Penrose inverse aims at generalising the inverse of a matrix such that it is also applicable to singular matrices [83].

Theorem 3.8 (Discrete Green's Functions and Moore–Penrose Inverse)**.** *Let $\boldsymbol{v}$ denote the eigenvector to the singular eigenvalue of $\boldsymbol{L}$. If the discrete Green's functions $\boldsymbol{g}_{k,\ell}$ satisfy the additional constraint*

$$\langle \boldsymbol{v}, \boldsymbol{g}_{k,\ell} \rangle = 0 \qquad \textit{for all } (k,\ell) \in \Gamma, \tag{3.22}$$

then they are given by the columns of the Moore–Penrose inverse of $\boldsymbol{L}$.

Proof. To verify that $\boldsymbol{G}$ is the Moore–Penrose inverse of $\boldsymbol{L}$, we have to check the following properties (cf. [83]):

(i) $\boldsymbol{LGL} = \boldsymbol{L}$

(ii) $\boldsymbol{GLG} = \boldsymbol{G}$

(iii) $\boldsymbol{LG}$ is symmetric.

(iv) $\boldsymbol{GL}$ is symmetric.

Since $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{L}$ to the eigenvalue 0, we have

$$\boldsymbol{v}^\top \boldsymbol{L} = \boldsymbol{0}^\top. \tag{3.23}$$

Thus, together with (3.21), it follows that

$$\boldsymbol{LGL} = \left(\boldsymbol{I} - \frac{1}{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} (\boldsymbol{v})(\boldsymbol{v})^\top \right) \boldsymbol{L} = \boldsymbol{L} \tag{3.24}$$

and

$$\boldsymbol{GLG} = \boldsymbol{G} - \frac{1}{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} \boldsymbol{G}(\boldsymbol{v})(\boldsymbol{v})^\top. \tag{3.25}$$

The condition $\langle \boldsymbol{v}, \boldsymbol{g}_{k,\ell} \rangle = 0$ implies $\boldsymbol{G}(\boldsymbol{v}) = \boldsymbol{0}$, and hence

$$\boldsymbol{GLG} = \boldsymbol{G}. \tag{3.26}$$

From (3.21) it is evident that $\boldsymbol{LG}$ is symmetric. We further compute:

$$\boldsymbol{GL} = (\boldsymbol{L}^\top \boldsymbol{G}^\top)^\top = (\boldsymbol{LG})^\top = \boldsymbol{LG} = (\boldsymbol{G}^\top \boldsymbol{L}^\top)^\top = (\boldsymbol{GL})^\top \tag{3.27}$$

Here, the symmetry of $\boldsymbol{L}$ and $\boldsymbol{G}$ is used, as well as property (iii) which has already be proven. Thus, $\boldsymbol{GL}$ is symmetric, too. □

Remark 3.9. Note that in [3], where Theorem 3.8 has been presented, there is a mistake in the proof of property (iv): The hypothesis of the theorem has been used to verify the symmetry of $\boldsymbol{GL}$, which is not valid. Hence, an adapted correct proof is presented above.

3.3.3 Construction of Discrete Green's Functions

After getting to know the theory behind discrete Green's functions, we will investigate how to practically construct them for the inpainting operators presented in Section 2.3. To this end, it is important to stress that all the operators $\boldsymbol{A}_N, \boldsymbol{B}_N, \boldsymbol{A}_P$ and $\boldsymbol{B}_P$ have a zero eigenvalue $\lambda_{0,0}$ belonging to the constant eigenvector $\boldsymbol{v}_{0,0}$ with entries $1/\sqrt{MN}$. Thus, we know from Definition 3.4 that in a point $(k,\ell) \in \Gamma$, the Green's function $\boldsymbol{g}_{k,\ell}$ for all operators are not unique. It satisfies the following system of equations:

$$(\boldsymbol{L}\boldsymbol{g}_{k,\ell})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} - \frac{1}{MN} \qquad \text{for } (i,j) \in \Gamma, \tag{3.28}$$

with $\boldsymbol{L} \in \{\boldsymbol{A}_N, \boldsymbol{B}_N, \boldsymbol{A}_P, \boldsymbol{B}_P\}$.

Obviously, one possibility to find the discrete Green's function is to solve the linear system of equations directly. However, this is very cumbersome, especially when one wants to seek a large number of discrete Green's functions. Even an iterative scheme does not allow to compute a sufficiently accurate discrete Green's function in a reasonable amount of time for our purposes.

For this reason, it would be nice to have a closed form solution of the discrete Green's functions. In this context, there exist some designated approaches for specific problem settings [135]. However, as the Green's functions themselves, most of these techniques are considered in a continuous setting. The probably most promising technique is the so-called *method of eigenfunction expansion* [135] which can also be applied in the discrete setting [24]. The idea is to express the discrete Green's functions in terms of the eigenvectors and corresponding eigenvalues of the discrete operator $\boldsymbol{L}$. With this strategy we can obtain the following result:

Theorem 3.10 (Closed Form Solution of Discrete Green's Functions)**.** *In a point $(k,\ell) \in \Gamma$ the discrete Green's functions for the matrix $\boldsymbol{L} \in \{\boldsymbol{A}_N, \boldsymbol{B}_N, \boldsymbol{A}_P, \boldsymbol{B}_P\}$ are given by*

$$(\boldsymbol{g}^c_{k,\ell})_{i,j} = \sum_{\substack{m=0 \\ (m,n)\neq(0,0)}}^{M-1} \sum_{n=0}^{N-1} \left[\frac{1}{\lambda_{m,n}} \cdot (\boldsymbol{v}_{m,n})_{k,\ell} \cdot (\boldsymbol{v}_{m,n})_{i,j} \right] + c, \tag{3.29}$$

where $\lambda_{m,n}$ are the eigenvalues corresponding to the eigenvectors $\boldsymbol{v}_{m,n}$ of the respective *operator $\boldsymbol{L}$, and the constant $c \in \mathbb{R}$ can be chosen arbitrarily.*

Proof. Following [24], we express the Green's function in terms of the orthonormal eigenvectors:

$$\boldsymbol{g}_{k,\ell} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} c_{m,n} \boldsymbol{v}_{m,n} \tag{3.30}$$

with coefficients $c_{m,n} \in \mathbb{R}$. Plugging this into (3.28) yields

$$\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} c_{m,n} \lambda_{m,n} (\boldsymbol{v}_{m,n})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} - \frac{1}{MN}. \tag{3.31}$$

After multiplying both sides with $(\boldsymbol{v}_{m',n'})_{i,j}$ for fixed $(m', n') \in \Gamma$, and summing up over all pixels $(i, j) \in \Gamma$, we have

$$c_{m',n'} \lambda_{m',n'} \;=\; (\boldsymbol{v}_{m',n'})_{k,\ell} - \frac{1}{MN} \sum_{(i,j)\in\Gamma} (\boldsymbol{v}_{m',n'})_{i,j}. \tag{3.32}$$

For $m' = n' = 0$, the eigenvalue $\lambda_{0,0}$ as well as the right hand side become 0. Thus, $c_{0,0}$ can be chosen arbitrarily. This means that the Green's function is unique up to a constant c. For $m' > 0$ or $n' > 0$, we notice that the sum vanishes due to the fact that the eigenvectors are orthogonal to the constant eigenvector. Thus, we obtain

$$c_{m,n} = \frac{1}{\lambda_{m,n}} \left(\boldsymbol{v}_{m,n}\right)_{k,\ell}. \tag{3.33}$$

This concludes the proof. □

We specify a canonic representative $\boldsymbol{g}^0_{k,\ell}$ by setting the constant $c := 0$. As the eigenvectors $\boldsymbol{v}_{m,n}$ with $(m, n) \neq (0, 0)$ of the discrete operator are orthogonal to $\boldsymbol{v}_{0,0}$, this is equivalent to assuming $\langle \boldsymbol{g}^0_{k,\ell}, \boldsymbol{v}_{0,0} \rangle = 0$. This shows that the obtained Green's functions $\boldsymbol{g}^0_{k,\ell}$ have mean value zero. Example plots of Green's functions are depicted in Figure 3.1 and 3.2.

3.3.4 Efficient Computation of Inner Products

In many practical applications, it is necessary to compute the inner products of the discrete Green's functions with a fixed image. For example, this becomes obvious with the following Theorem:

Theorem 3.11 (Solution of Discrete Problem via Inner Products)**.** *The solution of the discrete problem $\boldsymbol{Lu} = \boldsymbol{a}$ is given at a position $(i, j) \in \Gamma$ by $u_{i,j} = \left\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \right\rangle$.*

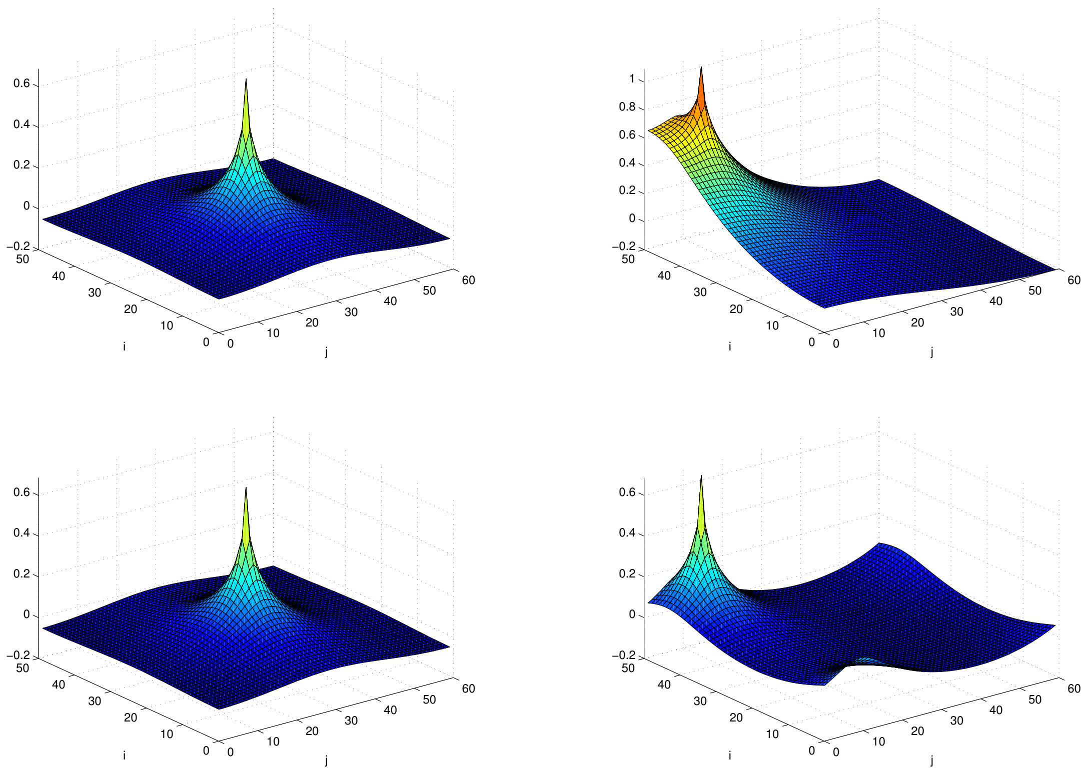

Figure 3.1. Example discrete Green's functions for Laplacian $\boldsymbol{A}_P$ on an image of size 50×60. **Left column:** $\boldsymbol{g}^{\mathbf{0}}_{\mathbf{25,30}}$. **Right column:** $\boldsymbol{g}^{\mathbf{0}}_{\mathbf{45,10}}$. **Top row:** Homogeneous Neumann boundary conditions. **Bottom row:** Periodic boundary conditions.

Proof. Recall from Section 3.3 that the solution of the discrete problem $\boldsymbol{L}\boldsymbol{u} = \boldsymbol{a}$ is given at a position $(i, j) \in \Gamma$ by

$$u_{i,j} = \sum_{(k,\ell)\in\Gamma} a_{k,\ell} \, (\boldsymbol{g}^0_{k,\ell})_{i,j} . \tag{3.34}$$

We can reformulate this by employing the symmetry of the discrete Green's functions, see Theorem 3.7, and obtain:

$$u_{i,j} = \sum_{(k,\ell)\in\Gamma} a_{k,\ell} \, (\boldsymbol{g}^0_{i,j})_{k,\ell} = \left\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \right\rangle . \tag{3.35}$$

□

The preceding theorem shows that an efficient computation of the inner products is very important as it directly leads to the solution of the Poisson equation for example. In the following, we will thus investigate how the inner products can be obtained in a fast way. The findings in the present Section are based on unpublished ideas and notes

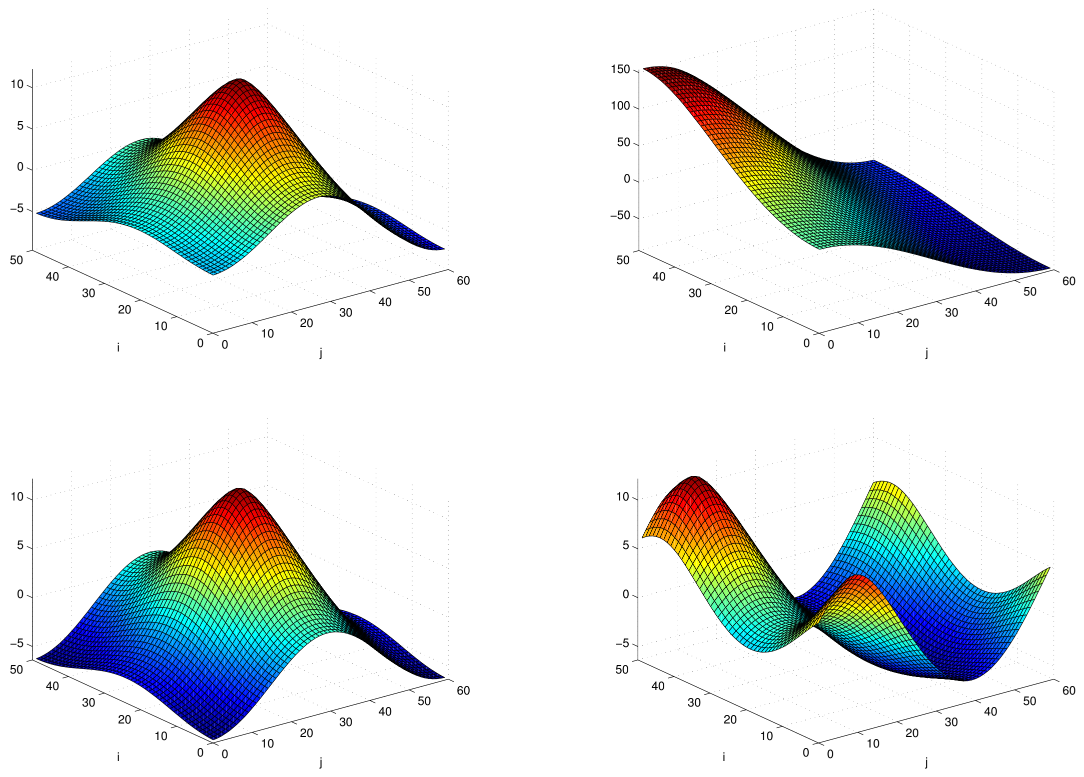

Figure 3.2. Example discrete Green's functions for the biharmonic operator $\boldsymbol{B}_P$ on an image of size 50×60. **Left column:** $\boldsymbol{g}^{\boldsymbol{0}}_{\boldsymbol{25,30}}$. **Right column:** $\boldsymbol{g}^{\boldsymbol{0}}_{\boldsymbol{45,10}}$. **Top row:** Homogeneous Neumann boundary conditions. **Bottom row:** Periodic boundary conditions.

of Prof. Gerlind Plonka-Hoch, for which I explicitly want to express my gratitude.

Considering the inner product $\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \rangle$, we can insert the analytic expression of the Green's functions (cf. Theorem 3.10) and obtain:

$$\begin{aligned}
\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \rangle &= \sum_{(k,\ell)\in\Gamma} a_{k,\ell} \, (\boldsymbol{g}^0_{i,j})_{k,\ell} \\
&= \sum_{(k,\ell)\in\Gamma} a_{k,\ell} \cdot \left[\sum_{\substack{m=0 \\ (m,n)\neq(0,0)}}^{M-1} \sum_{n=0}^{N-1} \left(\frac{1}{\lambda_{m,n}} \cdot (\boldsymbol{v}_{m,n})_{k,\ell} \cdot (\boldsymbol{v}_{m,n})_{i,j} \right) \right] \\
&= \sum_{\substack{m=0 \\ (m,n)\neq(0,0)}}^{M-1} \sum_{n=0}^{N-1} \left[(\boldsymbol{v}_{m,n})_{i,j} \cdot \frac{1}{\lambda_{m,n}} \cdot \underbrace{\sum_{(k,\ell)\in\Gamma} \left(a_{k,\ell} \cdot (\boldsymbol{v}_{m,n})_{k,\ell} \right)}_{:=\widehat{a}_{m,n}} \right] . \qquad (3.36)
\end{aligned}$$

Algorithm 3.1. Fast Computation of Inner Products and Solution of Discrete Problem

Input: Image $\boldsymbol{a}$.

Compute:

1. Transform $\boldsymbol{a}$ to frequency domain using the appropriate basis functions given by the eigenvectors of the operator to obtain $\widehat{\boldsymbol{a}}$.
2. Obtain $\widehat{\boldsymbol{a}}'$ with Equation 3.37.
3. Apply the inverse frequency transform on $\widehat{\boldsymbol{a}}'$.

Output: Inner products and solution $\boldsymbol{u}$ of discrete problem $\boldsymbol{L}\boldsymbol{u} = \boldsymbol{a}$ as $u_{i,j} = \left\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \right\rangle$ for all $(i,j) \in \Gamma$.

As one can see, the expression $\widehat{a}_{m,n}$ only has to be computed once for all (m,n). The resulting image $\widehat{\boldsymbol{a}}$ can then be reused to obtain the inner products for all $(i,j) \in \Gamma$. Recall that the eigenvectors of the operators with homogeneous Neumann boundary conditions are given by the basis vectors of the discrete cosine transform. Thus, the term $\widehat{\boldsymbol{a}}$ can be interpreted as a basis transform into the cosine frequency domain. There exist several methods to efficiently compute this discrete cosine transform [159]. Similarly, for periodic boundary conditions, the Fast Fourier Transform (FFT) can be employed to efficiently compute the transform of $\boldsymbol{a}$ to the frequency domain [57]. Especially when we have an image grid which has a size of powers of 2, the FFT algorithm performs very fast.

Based on $\widehat{\boldsymbol{a}}$, we define a modified version $\widehat{\boldsymbol{a}}'$ of it:

$$\widehat{a}'_{m,n} = \begin{cases} \frac{1}{\lambda_{m,n}} \widehat{a}_{m,n} & \text{if } (m,n) \neq (0,0), \\ 0 & \text{if } (m,n) = (0,0). \end{cases} \tag{3.37}$$

This new term allows to write the inner product as

$$\left\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{a} \right\rangle = \sum_{(m,n)\in\Gamma} (\boldsymbol{v}_{m,n})_{i,j} \cdot \widehat{a}'_{m,n} = \sum_{(m,n)\in\Gamma} (\boldsymbol{v}_{i,j})_{m,n} \cdot \widehat{a}'_{m,n} \,. \tag{3.38}$$

Analogously, this can be interpreted as a back transformation from the frequency domain to the spatial domain using the same basis vectors as before. To this end, the same methods as for example the FFT can be employed to efficiently obtain the inner products of all discrete Green's functions with $\boldsymbol{a}$. Besides these inner products, we know from Theorem 3.11 that the very same procedure leads to the solution of the discrete problem $\boldsymbol{L}\boldsymbol{u} = \boldsymbol{a}$. Algorithm 3.1 summarises the overall procedure.

Related Fourier approaches without discrete Green's functions are known in the literature from the field of numerical partial differential equations under the name *Fast Poisson Solvers* [184, 193]. Thereby, the strategy is to express the solution $\boldsymbol{u}$

directly in terms of the corresponding eigenvectors.

3.3.5 Efficient Computation of Green's Functions

In Section 3.3.3, we have found a nice representation of the discrete Green's functions. However, computing the double summation for obtaining the Green's functions is very time consuming. Especially when one is interested in computing the Green's functions for a large number of points or even all grid points - which is often the case in practical scenarios - this can take several hours or even days depending on the image size. To tackle this issue, we propose different strategies in the following which allow us to obtain the individual Green's functions in a much faster way.

Symmetry of Image Domain

First of all, we can exploit the symmetry of the rectangular image domain to reduce the effort for computing all discrete Green's functions by a factor of 4: Once the Green's function is computed for a specific source point $(k, \ell) \in \Gamma$, the Green's functions for the source points $(M-k, \ell)$, $(k, N-\ell)$, and $(M-k, N-\ell)$ can be obtained by mirroring $\boldsymbol{g}^0_{k,\ell}$ along the x axis, the y axis and both axes, respectively.

Symmetry of Discrete Green's Functions

Secondly, we can derive another speed-up strategy from the symmetry of the discrete Green's functions, cf. Theorem 3.7: After having computed one Green's function for a specific source point $(k, \ell) \in \Gamma$, we know the value of all Green's functions at the position (k, ℓ). This allows to further reduce the computational effort. For example, when one wants to obtain all discrete Green's functions for MN pixel locations, one does not have to compute the values for $(MN)^2$ pixels, but only for $\frac{(MN)\cdot(MN+1)}{2}$ pixels. This means we gain a speed-up factor of around 2.

Green's Function in Frequency Domain

The considerations in Section 3.3.4 allow to come up with a very efficient way to compute a discrete Green's function. To this end, let us write the Green's function as an inner product with the Kronecker delta function:

$$(\boldsymbol{g}^0_{k,\ell})_{i,j} = \left\langle \boldsymbol{g}^0_{k,\ell}, \boldsymbol{\delta}_{i,j} \right\rangle = \left\langle \boldsymbol{g}^0_{i,j}, \boldsymbol{\delta}_{k,\ell} \right\rangle. \tag{3.39}$$

From (3.36) we know that the inner product can be obtained by a transformation of $\boldsymbol{\delta}_{k,\ell}$ into a frequency domain. This frequency domain depends on the eigenvectors $\boldsymbol{v}_{m,n}$ of the operator. This leads to the transformed Kronecker delta function $\widehat{\boldsymbol{\delta}_{k,\ell}}$:

$$(\widehat{\boldsymbol{\delta}}_{k,\ell})_{m,n} = \sum_{(i,j)\in\Gamma} (\boldsymbol{\delta}_{k,\ell})_{i,j} (\boldsymbol{v}_{m,n})_{i,j} = (\boldsymbol{v}_{m,n})_{k,\ell}. \tag{3.40}$$

Algorithm 3.2. Generation of Discrete Green's Function from Frequency Domain

```
Input: Source point $(k,\ell) \in \Gamma$.
Compute:
    1. Obtain $\widehat{\boldsymbol{\delta}}'_{k,\ell}$ given in (3.41).
    2. Compute back transformation of $\widehat{\boldsymbol{\delta}}'_{k,\ell}$ from frequency domain to spatial domain.
Output: Discrete Green's functions $\boldsymbol{g}^0_{k,\ell}$.
```

As a result, the modified version $\widehat{\boldsymbol{\delta}}'_{k,\ell}$ of $\widehat{\boldsymbol{\delta}}_{k,\ell}$ as described in Equation 3.37 can be written down explicitly:

$$(\widehat{\boldsymbol{\delta}}'_{k,\ell})_{m,n} = \begin{cases} \frac{1}{\lambda_{m,n}} (\boldsymbol{v}_{m,n})_{k,\ell} & \text{if } (m,n) \neq (0,0), \\ 0 & \text{if } (m,n) = (0,0). \end{cases} \tag{3.41}$$

Note that $\widehat{\boldsymbol{\delta}}'_{k,\ell}$ can be considered as the discrete Green's function in the frequency domain. The sought discrete Green's function is thus simply given as a back transformation of $\widehat{\boldsymbol{\delta}}'_{k,\ell}$ to the spatial domain:

$$\boldsymbol{g}^0_{k,\ell} = \sum_{(m,n)\in\Gamma} (\boldsymbol{v}_{i,j})_{m,n} \cdot (\widehat{\boldsymbol{\delta}}'_{k,\ell})_{m,n} \, . \tag{3.42}$$

As discussed in Section 3.3.4, there are very fast algorithms such as the FFT to perform this back transformation. Thus, we are able to compute a discrete Green's function in an efficient way. An overview over the complete procedure is given in Algorithm 3.2. As an example, for an image of size 512×512, the computation of one Green's function in a naive way by computing the double summation in Section 3.3.3 requires 60 seconds. The presented new approach only needs 0.03 seconds. This shows that we have gained a speed-up by a factor of 2000. Although 0.03 seconds might seem as a very short time, note that the computation of a large number of discrete Green's functions still requires a lot of time. In practice it is common that one has to compute tens of thousands of Green's functions, leading to runtimes of several minutes.

Generation of Discrete Green's Functions

A further strategy is to generate all discrete Green's functions from only one precomputed reference Green's function. Thereby, the boundary conditions play an important role. I would like to thank Prof. Joachim Weickert for contributing this significant idea which is responsible for additional tremendous accelerations.

For the moment, let us consider periodic boundary conditions. We will see that Green's functions for this type of boundaries allow to come up with an efficient strategy to generate Green's functions with homogeneous Neumann boundary conditions.

Periodic boundary conditions have the advantage that they inherently possess a recurring extension that allows for a simple trick: Given a reference Green's function $\boldsymbol{g}^0_{0,0}$ at the pixel position $(0,0) \in \Gamma$, one can generate all other Green's functions with periodic boundary conditions by simply shifting the reference Green's function to the respective source points:

Theorem 3.12 (Generation of Discrete Green's Functions for Periodic Boundary Conditions)**.** *Let $\boldsymbol{g}^0_{0,0}$ be the discrete Green's function for an operator with periodic boundary conditions and the source point* $(0,0)$. *Then the discrete Green's function $\boldsymbol{g}^0_{k,\ell}$ for this operator at a position $(i,j) \in \Gamma$ is given by*

$$(\boldsymbol{g}^0_{k,\ell})_{i,j} = (\boldsymbol{g}^0_{0,0})_{(k-i) \bmod M,\, (\ell-j) \bmod N} \, . \tag{3.43}$$

Proof. Assuming a periodic extension of $\boldsymbol{g}^0_{0,0}$, we can write

$$(\boldsymbol{g}^0_{0,0})_{(k-i) \bmod M,\, (\ell-j) \bmod N} = (\boldsymbol{g}^0_{0,0})_{k-i,\, \ell-j} \, . \tag{3.44}$$

Shifting the grid by i and j in x and y direction again leads to a periodic Green's function on Γ, with a source point now being at (k,ℓ). $\square$

This theorem allows us to instantly access all discrete Green's functions for an operator with periodic boundary conditions from only one precomputed Green's function. Of course, choosing the source point $(0,0)$ for the reference Green's function is arbitrary. We could also select any other point for this purpose and adapt the shifting accordingly.

When it comes to homogeneous Neumann boundary conditions, we cannot obtain the discrete Green's function by a pure shift. However, we can utilise the periodic Green's functions for this purpose. Thereby, a specific periodic discrete Green's function again serves a reference function. We call this key ingredient of the strategy the *Green's mother function.* This name is inspired by the well known wavelet transforms, where the basis functions are obtained as rescaled and shifted versions of a so called mother wavelet [128]. Before we can define the Green's mother function, a so called *extended operator* has to be defined:

Definition 3.13 (Extended Operator)**.** Let an image grid of size $M \times N$ and an operator $\boldsymbol{L}$ incorporating homogeneous Neumann boundary conditions be given. Then, the extended operator $\boldsymbol{L}_{ext}$ of $\boldsymbol{L}$ is defined as the operator which implements the same differential operator as $\boldsymbol{L}$, with the difference that it incorporates periodic boundary conditions and is defined on a larger grid Γ_{ext} of size $2M \times 2N$.

We can now introduce the definition of the Green's mother function (GMF):

Definition 3.14 (Green's Mother Function (GMF))**.** The Green's mother function $\boldsymbol{G}$ of an operator $\boldsymbol{L}$ is the discrete Green's function $\boldsymbol{g}^0_{0,0}$ of the extended operator $\boldsymbol{L}_{ext}$ of $\boldsymbol{L}$.

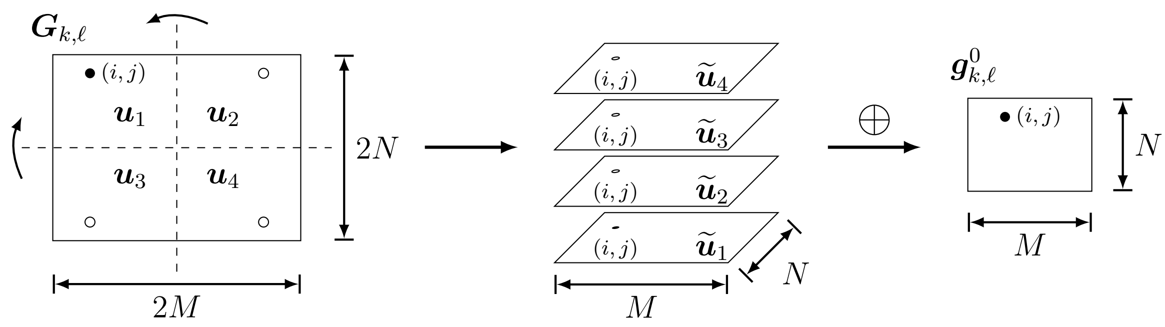

Figure 3.3. Sketch of generation of a discrete Green's function $\boldsymbol{g}_{k,\ell}$ for homogeneous Neumann boundary conditions from a Green's function $\boldsymbol{G}_{k,\ell}$ with periodic boundary conditions on a larger grid.

With the help of the GMF we can generate all discrete Green's functions for an operator $\boldsymbol{L}$ with homogeneous Neumann boundary conditions with only a few basic operations. This works as follows. In a first step, we employ Theorem 3.12 to obtain the discrete Green's function $\boldsymbol{G}_{k,\ell}$ of the extended operator $\boldsymbol{L}_{ext}$ of $\boldsymbol{L}$ for the source point $(k,\ell) \in \Gamma$. Note that $\boldsymbol{G}_{k,\ell}$ is defined on the large grid Γ_{ext}. This Green's function is processed further to obtain the sought Green's function. A sketch of this generation procedure is depicted in Figure 3.3. On the left, the image containing the discrete Green's function $\boldsymbol{G}_{k,\ell}$ is shown. The image domain is split into 4 parts, yielding the smaller images $\boldsymbol{u}_1,\ldots,\boldsymbol{u}_4$. By mirroring these images along the centre in x and y direction, respectively, one can obtain an alignment of the four depicted points corresponding to the point (i,j). This leads to the four new images $\widetilde{\boldsymbol{u}}_1,\ldots,\widetilde{\boldsymbol{u}}_4$ which live on the original grid Γ. After a pixelwise summation over these four images one obtains a final image $\boldsymbol{g}^0_{k,\ell}$. More precisely, this image is computed at a position $(i,j) \in \Gamma$ as

$$\begin{aligned}(\boldsymbol{g}^0_{k,\ell})_{i,j} &:= (\widetilde{\boldsymbol{u}}_1)_{i,j} + (\widetilde{\boldsymbol{u}}_2)_{i,j} + (\widetilde{\boldsymbol{u}}_3)_{i,j} + (\widetilde{\boldsymbol{u}}_4)_{i,j} \\ &= (\boldsymbol{u}_1)_{i,j} + (\boldsymbol{u}_2)_{M-1-i,j} + (\boldsymbol{u}_3)_{i,N-1-j} + (\boldsymbol{u}_4)_{M-1-i,N-1-j} \\ &= (\boldsymbol{G}_{k,l})_{i,j} + (\boldsymbol{G}_{k,l})_{2M-1-i,j} + (\boldsymbol{G}_{k,l})_{i,2N-1-j} + (\boldsymbol{G}_{k,l})_{2M-1-i,2N-1-j}\,. \end{aligned} \tag{3.45}$$

As it turns out, this image $\boldsymbol{g}_{k,\ell}$ is the desired discrete Green's function for the operator $\boldsymbol{L}$ with homogeneous Neumann boundary conditions. This will be shown in the following theorem.

Theorem 3.15 (Generation of Discrete Green's Functions for Homogeneous Neumann Boundary Conditions)**.** *The obtained image $\boldsymbol{g}^0_{k,\ell}$ in Equation 3.45 is the discrete Green's function corresponding to the operator $\boldsymbol{L}$ with homogeneous Neumann boundary conditions.*

Proof. We have to check that $\boldsymbol{g}^0_{k,\ell}$ fulfils Equation 3.28. We know that $\boldsymbol{G}_{k,\ell}$ is the discrete Green's function of the extended operator $\boldsymbol{L}_{ext}$ of $\boldsymbol{L}$ on the large grid Γ_{ext}. Thus, we know from (3.28) that the following holds true for $(i,j) \in \Gamma_{ext}$:

$$(\boldsymbol{L}_{ext}\boldsymbol{G}_{k,\ell})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} - \frac{1}{2M \cdot 2N} \,. \tag{3.46}$$

With this, we compute for $(i,j) \in \Gamma$:

$$\begin{aligned}(\boldsymbol{L}\boldsymbol{g}^0_{k,\ell})_{i,j} &= (\boldsymbol{L}(\widetilde{\boldsymbol{u}}_1 + \widetilde{\boldsymbol{u}}_2 + \widetilde{\boldsymbol{u}}_3 + \widetilde{\boldsymbol{u}}_4))_{i,j} \\ &= (\boldsymbol{L}\widetilde{\boldsymbol{u}}_1)_{i,j} + (\boldsymbol{L}\widetilde{\boldsymbol{u}}_2)_{i,j} + (\boldsymbol{L}\widetilde{\boldsymbol{u}}_3)_{i,j} + (\boldsymbol{L}\widetilde{\boldsymbol{u}}_4)_{i,j} \\ &= (\boldsymbol{L}\boldsymbol{u}_1)_{i,j} + (\boldsymbol{L}\boldsymbol{u}_2)_{M-1-i,j} + (\boldsymbol{L}\boldsymbol{u}_3)_{i,N-1-j} + (\boldsymbol{L}\boldsymbol{u}_4)_{M-1-i,N-1-j} \,. \end{aligned} \tag{3.47}$$

The individual images $\boldsymbol{u}_1, \dots, \boldsymbol{u}_4$ can be extended to the large grid Γ_{ext} by mirroring them along the two axes similar to the sketch in Figure 3.3. Let us denote the resulting extended images by $\boldsymbol{U}_1, \dots, \boldsymbol{U}_4$. Note that by construction these extended images inherently fulfil both homogeneous Neumann boundary conditions at the mirror axes as well as periodic boundary conditions at the boundaries of the extended grid. Thus, we can replace the operator $\boldsymbol{L}$ by its extended variant and compute:

$$\begin{aligned}(\boldsymbol{L}\boldsymbol{g}^0_{k,\ell})_{i,j} &= (\boldsymbol{L}_{ext}\boldsymbol{U}_1)_{i,j} + (\boldsymbol{L}_{ext}\boldsymbol{U}_2)_{i,j} + (\boldsymbol{L}_{ext}\boldsymbol{U}_3)_{i,j} + (\boldsymbol{L}_{ext}\boldsymbol{U}_4)_{i,j} \\ &= (\boldsymbol{L}_{ext}\boldsymbol{G}_{k,\ell})_{i,j} + (\boldsymbol{L}_{ext}\boldsymbol{G}_{k,\ell})_{2M-1-i,j} \\ &\quad + (\boldsymbol{L}_{ext}\boldsymbol{G}_{k,\ell})_{i,2N-1-j} + (\boldsymbol{L}_{ext}\boldsymbol{G}_{k,\ell})_{2M-1-i,2N-1-j} \\ &= (\boldsymbol{\delta}_{k,\ell})_{i,j} + (\boldsymbol{\delta}_{k,\ell})_{2M-1-i,j} + (\boldsymbol{\delta}_{k,\ell})_{i,2N-1-j} \\ &\quad + (\boldsymbol{\delta}_{k,\ell})_{2M-1-i,2N-1-j} - \frac{4}{4MN} \,. \end{aligned} \tag{3.48}$$

We know that $(k,\ell) \in \Gamma$. Thus, the Kronecker delta function can only be non-zero for one Kronecker delta function, which leads to:

$$(\boldsymbol{L}\boldsymbol{g}^0_{k,\ell})_{i,j} = (\boldsymbol{\delta}_{k,\ell})_{i,j} - \frac{1}{MN} \tag{3.49}$$

As we can see, the generated image $\boldsymbol{g}^0_{k,\ell}$ fulfils the desired linear system of equations. Thus, it is the desired discrete Green's function. □

To summarise, all the discrete Green's functions for homogeneous Neumann boundary conditions can be generated from the GMF - the Green's mother function. This is achieved with a few basic operations which can be performed very fast. The GMF only has to be computed once in the beginning, which can be done with the strategy presented in Section 3.3.5.

A summary over all individual steps on how to obtain the discrete Green's functions for homogeneous Neumann boundary conditions from the GMF is shown in Algorithm 3.3. With this strategy, one can generate the discrete Green's functions for

Algorithm 3.3. Generation of Discrete Green's Function for Homogeneous Neumann Boundary Conditions from the GMF

```
Input: Source point $(k,\ell) \in \Gamma$, Green's mother function $\boldsymbol{G}$.
Compute:
    1. Generate $\boldsymbol{G}_{k,\ell}$ by shifting $\boldsymbol{G}$ to the position $(k,\ell)$ using Theorem 3.12.
    2. Compute $\boldsymbol{g}^0_{k,\ell}$ from $\boldsymbol{G}_{k,\ell}$ with (3.45).
Output: Discrete Green's functions $\boldsymbol{g}^0_{k,\ell}$.
```

all pixels in a matter of seconds.

3.4 Inpainting with Green's Functions

Using the presented theory, it is possible to describe the solution of the discrete inpainting problem in terms of the discrete Green's functions. To do so, we rewrite the inpainting problem in Equations (2.10) and (2.11) such that it has the form as in (3.12). Thereby, the right hand side $\boldsymbol{a}$ is constructed such that it is zero at all non-mask points, while its values at all mask points $(i,j) \in K$ are unknown and have to be determined later. The new inpainting problem reads as

$$\boldsymbol{L}\boldsymbol{u} = \boldsymbol{a}\,, \tag{3.50}$$

subject to

$$u_{i,j} = f_{i,j} \qquad \text{if } (i,j) \in K\,, \tag{3.51}$$

$$a_{i,j} = 0 \qquad \text{if } (i,j) \in \Gamma \backslash K\,. \tag{3.52}$$

We know that the operator $\boldsymbol{L}$ has a singular eigenvalue with corresponding constant eigenvector $\boldsymbol{v}_{0,0}$. Thus, we know from the Fredholm alternative that the solutions of (3.50) only exist if the solvability condition (3.17) is fulfilled:

$$\langle \boldsymbol{v}_{0,0},\, \boldsymbol{a} \rangle = 0 \qquad \Longleftrightarrow \qquad \sum_{(k,\ell)\in K} a_{k,\ell} = 0, \tag{3.53}$$

If this is the case, the solutions are given as

$$u_{i,j} = \sum_{(k,\ell)\in\Gamma} a_{k,\ell} \cdot (\boldsymbol{g}^0_{k,\ell})_{i,j} + c\,, \tag{3.54}$$

with an unknown constant $c \in \mathbb{R}$, comprising all constants of the individual Green's functions. The constraint (3.52) leads to vanishing entries of $\boldsymbol{a}$ at all non-mask points.

The solution can thus be restricted to the set of all mask points:

$$u_{i,j} = \sum_{(k,\ell)\in K} a_{k,\ell} \cdot (\boldsymbol{g}^0_{k,\ell})_{i,j} + c\,. \tag{3.55}$$

This representation shows that the inpainting solution can be composed by a small number of atoms, namely the discrete Green's functions corresponding to the mask pixels. In other words, the discrete Green's functions $\boldsymbol{g}^0_{k,\ell}$ corresponding to $(k,\ell) \in K$ can be seen as a generating system for the space of all inpainting solutions on Γ with zero mean. This enables us to connect PDE based inpainting ideas with the concept of sparsity.

Together with the solvability condition (3.53), Equation 3.51 specifies the remaining unknown coefficients c and $a_{k,\ell}$, $(k,\ell) \in K$, and hence the inpainting result uniquely. Denoting the 2D pixel indices of the mask points by $m_1,\ldots,m_L$, with $L := |K|$, we can set up the linear system of equations for finding the unknown values of $\boldsymbol{a}$ and c:

$$\begin{pmatrix} (\boldsymbol{g}^0_{m_1})_{m_1} & (\boldsymbol{g}^0_{m_2})_{m_1} & \cdots & (\boldsymbol{g}^0_{m_L})_{m_1} & 1 \\ (\boldsymbol{g}^0_{m_1})_{m_2} & (\boldsymbol{g}^0_{m_2})_{m_2} & \cdots & (\boldsymbol{g}^0_{m_L})_{m_2} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ (\boldsymbol{g}^0_{m_1})_{m_L} & (\boldsymbol{g}^0_{m_2})_{m_L} & \cdots & (\boldsymbol{g}^0_{m_L})_{m_L} & 1 \\ 1 & 1 & \ldots & 1 & 0 \end{pmatrix} \begin{pmatrix} a_{m_1} \\ a_{m_2} \\ \vdots \\ a_{m_L} \\ c \end{pmatrix} = \begin{pmatrix} f_{m_1} \\ f_{m_2} \\ \vdots \\ f_{m_L} \\ 0 \end{pmatrix}. \tag{3.56}$$

For solving this system of equations, we recommend the QR algorithm since it does not create error accumulations [165]. Once the values for c and $a_{m_1},\ldots,a_{m_L}$ are computed, the inpainting solution $\boldsymbol{u}$ is given exactly by (3.55). For this purpose, the discrete Green's functions can be obtained with the help of the Green's mother function and Theorem 3.15. As an alternative to the superposition, Algorithm 3.1 can be employed to obtain the solution of the reformulated discrete problem (3.50). Subsequently adding the constant c yields the inpainting solution. This approach has the advantage that it avoids evaluating the discrete Green's functions at non-mask locations. For the reader's convenience, Algorithm 3.4 summarises the complete procedure of the presented inpainting strategy.

As a remark, the mean value of the final inpainting is given by c, because the mean value of the discrete Green's functions equals 0 as shown earlier. Moreover, note that the coefficients $a_{m_1},\ldots,a_{m_L}$ directly correspond to the derivative information $\boldsymbol{L}\boldsymbol{u}$ of the inpainting. From Equation (3.54) we know that these coefficients together with the mean value c uniquely specify the inpainting. On the other hand, we know that we can uniquely describe an inpainting by prescribing tonal values at the mask points. Thus, we have found two fundamentally distinct possibilities to describe the very same inpainting by specifying different information at the mask locations.

A decisive advantage of our inpainting algorithm with Green's functions is that it reveals the influence of each mask point on the overall inpainting result as it is

Algorithm 3.4. Inpainting with Green's functions.

```
Input: Image $\boldsymbol{f}$ at specified mask $K$.
Compute:
    1. Obtain the Green's mother function as described in Section 3.3.5.
    2. Compute the Green's functions $\boldsymbol{g}^0_{k,\ell}$ for all $(k,\ell) \in K$ from the Green's mother
       function with Algorithm 3.3.
    3. Solve (3.56) to find the unknown coefficients of $\boldsymbol{a}$ and $c$ .
    4. Obtain the solution $\boldsymbol{u}$ as the superposition given in (3.55), or by employing
       Algorithm 3.1 and subsequently adding $c$.
Output: Inpainting solution $\boldsymbol{u}$.
```

described by the respective Green's function. Obviously, the complexity for finding a solution increases with the number of mask points. This means that this approach is faster if the specified mask becomes sparser. Interestingly, this is contrary to standard iterative approaches for solving the discrete inpainting problem. Here, a sparser mask usually leads to increased runtimes, because the diffusion of the prescribed data over larger parts of the image takes more time. Since one typically deals with sparse masks in image compression scenarios, Green's functions can thus help to improve the efficiency. As a remark, one has to mention that the discrete Green's functions so far are only defined for linear operators, and consequently the preceding reformulation of the inpainting problem is only valid for such inpainting operators.

3.5 Runtime Comparison

To evaluate the performance of the inpainting strategy with discrete Green's functions, the following experiment is conducted with a C implementation on a desktop PC with Intel Xeon processor (3.2GHz). For different masks of densities ranging from 0.01% to 16%, we compute the corresponding inpainting solutions for harmonic and biharmonic inpainting. Thereby, the grey values of the image *trui* are prescribed at the mask locations. The masks are the same as the ones in Section 2.5, i.e. they are obtained with the approach of Belhachmi *et al.*. For some density values those masks have already been shown in Figure 2.4. The runtimes for the full inpainting with discrete Green's functions is shown in Figure 3.4. Note that the required seconds for the inpaintings are plotted on a logarithmic scale. It is not surprising that the runtime is larger the more mask points are present as the system matrix for determining the coefficients of the discrete Green's functions becomes larger. Note however that the runtime is independent of the chosen inpainting operator.

For the sake of comparison the inpainting is also computed with a bidirectional multigrid method. However, one has to keep in mind one major difference between the two approaches: While the discrete Green's functions allow to obtain an analytic inpainting solution, the multigrid solver only yields an approximate solution. By choosing appropriate numbers of outer cycles and inner iterations in the multigrid approach, it is possible to steer the precision of the solution. However, the very exact solution can hardly be obtained by using such a numerical solver. Thus, in this experiment, the runtimes for different precisions are evaluated. Unfortunately, is is not obvious how to choose the optimal number of cycles and inner iterations automatically. In particular, it is hard to say if more cycles and less inner iterations are better than less cycles with more inner iterations. Thus, for this experiment, the parameters are optimised in the following way. Given a prescribed allowed MSE value e, the inpaintings for many different combinations of inner and outer cycles are computed. Then, it is checked for each inpainting $\boldsymbol{u}$ if it achieves the desired precision compared to the reference ground truth inpainting $\boldsymbol{u}^{\text{ref}}$ obtained by inpainting with the discrete Green's functions. This test is passed if $\text{MSE}(\boldsymbol{u}, \boldsymbol{u}^{\text{ref}}) \leq e$ and at the same time $\left|\text{MSE}(\boldsymbol{f}, \boldsymbol{u}) - \text{MSE}(\boldsymbol{f}, \boldsymbol{u}^{\text{ref}})\right| \leq e$, with $\boldsymbol{f}$ being the original image. Both tests are necessary to guarantee for a meaningful judgement of the inpainting quality. Among all the inpaintings which successfully pass the test, one chooses the corresponding parameter combination which allows the fastest inpainting. This procedure allows to select the optimal parameters of the multigrid algorithm for a certain precision. The resulting multigrid parameters is given in Table 3.1 for the harmonic inpainting and the corresponding results for the biharmonic operator are shown in Table 3.2.

The corresponding runtime results with the multigrid approach are also plotted in Figure 3.4. As one can see, the full inpainting relying on discrete Green's functions is faster compared to the multigrid approaches only for smaller mask densities and if high accuracies are desired. While the inpainting with discrete Green's functions is much faster for smaller densities, the multigrid approaches perform better for larger densities in general. For both harmonic and biharmonic inpainting, the multigrid approaches are able to yield an approximate solution very fast, assuming the inner and outer cycles are optimally chosen. While the Green's approach requires the same amount of time for both inpainting operators, the multigrid approach needs different times. In particular, especially for smaller densities and higher precisions, the multigrid is faster with the biharmonic operator compared to the harmonic operator, while less time is needed with the harmonic operator for lower precisions. In other words, the standard deviation of the inpainting time for different precisions with the multigrid approach is smaller for the biharmonic operator than for the harmonic operator.

In the full inpainting with the discrete Green's functions, the coefficients for the individual discrete Green's functions have to be computed first by solving (3.56) before the inpainting is obtained as the superposition of the discrete Green's functions. This first step is very time-consuming. Hence, it would be beneficial if one could leave this step away by assuming that the coefficients c and $a_{m_1}, \dots, a_{m_{L-1}}$ are already given. This could for instance be the case in a compression framework where those values are

Table 3.1. Number of cycles / inner iterations of fastest multigrid performances for homogeneous diffusion inpainting with different mask densities and precisions.

MSE \ density	0.01	0.5	1	2	4	8	16
1	3 / 5	3 / 2	3 / 2	3 / 2	2 / 2	1 / 3	1 / 2
0.1	4 / 8	4 / 2	4 / 3	4 / 2	4 / 2	3 / 2	1 / 3
0.01	5 / 11	4 / 14	5 / 5	4 / 14	5 / 2	4 / 2	3 / 2
0.001	5 / 88	6 / 5	5 / 21	5 / 17	5 / 8	5 / 3	3 / 5
0.0001	6 / 99	6 / 18	6 / 23	7 / 9	5 / 20	5 / 9	3 / 12

Table 3.2. Number of cycles / inner iterations of fastest multigrid performances for biharmonic inpainting with different mask densities and precisions.

MSE \ density	0.01	0.5	1	2	4	8	16
1	4 / 6	2 / 5	3 / 3	2 / 3	2 / 3	1 / 5	1 / 3
0.1	4 / 9	2 / 17	5 / 3	2 / 15	2 / 6	2 / 3	1 / 5
0.01	5 / 9	2 / 86	6 / 3	2 / 20	4 / 5	2 / 5	2 / 5
0.001	5 / 14	4 / 17	6 / 5	4 / 5	4 / 8	2 / 15	2 / 8
0.0001	5 / 21	4 / 52	6 / 8	3 / 30	4 / 11	3 / 8	4 / 3

stored instead of the tonal data. As we know from Section 3.4 both representations allow to recover the very same inpainting solution. Note that the amount of data which needs to be stored is the same. In particular, the missing coefficient a_{m_L} can be recovered with the help of the solvability condition (3.53).

Then, the inpainting is directly given as the solution of the discrete problem (3.50), which can be obtained in two different ways. The most straightforward way is to assemble it as superposition of the discrete Green's functions according to Equation (3.55). In Figure 3.4 this approach is denoted as the 'assembled Green's functions'. As one can see, the time for the naïve computation of the superposition linearly depends on the density.

An alternative way to obtain the solution from the coefficients without explicitly accessing any Green's function is to employ Algorithm 3.1. This allows to obtain the inpainting in constant time with the help of the fast computation of the inner products in the frequency domain. This is also demonstrated in Figure 3.4: Independent of the mask density, it constantly requires only around 0.05 seconds to yield the inpainting solution, thus allowing for a real-time decoding on the CPU. If one is interested in at least a small amount of accuracy, this approach the most preferable one from all presented algorithms, assuming one is free to choose the data to be stored.

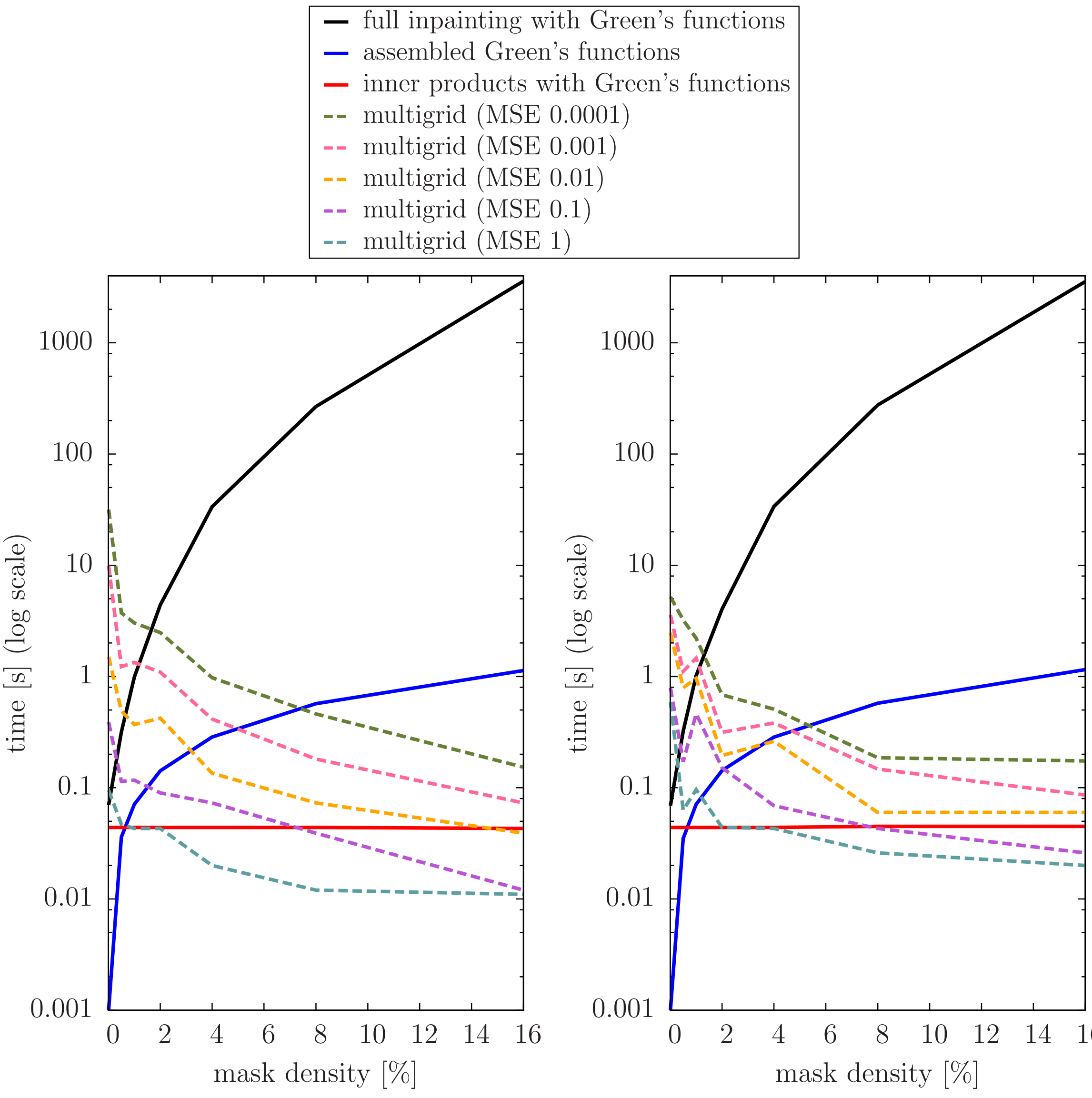

Figure 3.4. Runtime comparison of different inpainting strategies for various mask densities. **Left:** Harmonic inpainting. **Right:** Biharmonic inpainting.

In summary, there are two decisive advantages of the inpainting with discrete Green's functions compared to the multigrid approach. First of all, the complete approach is parameter-free. In particular, no adaptations need to be done for different mask densities or operators. The multigrid approach is only performing well when those parameters are chosen in a good way. One could think of learning those parameters, but this is very cumbersome. In practice, one thus often fixes them to some values which perform good in most applications. Another benefit of the approach with the discrete Green's functions lies in its enormous speed if one is free to store the coefficients instead of the tonal data. In that case, it yields the precise solution up to machine precision in constant time independent of the mask density or the inpainting operator. In particular, it is able to outperform the multigrid approach for small densities and even for larger densities if a higher accuracy is desired. Moreover, it is straightforward to extend the inpainting with discrete Green's functions for higher order inpainting operators by adapting the power of the eigenvalues accordingly. Everything else remains the same. In contrast, it requires much more effort to enable higher order operators in the multigrid approach.

3.6 Conclusion

Since one decade, the paradigms of sparse signal processing and inpainting methods for compact image representations have been enjoying a successful development. Although they often pursue similar goals, it is surprising that this has happened without any interaction.

The key concept for understanding this relation was the notion of discrete Green's functions. They serve as atoms in a dictionary. Only a single atom is needed to describe the global influence of one mask pixel. This allows to reinterpret successful inpainting methods with linear differential operators in terms of sparsity. Moreover, discrete Green's functions also offer an interesting interpretation as columns of the Moore–Penrose pseudo-inverse of the discretised (singular) differential operator.

The framework is fairly general: It is directly applicable to any linear self-adjoint differential operator with a known eigendecomposition. We have illustrated this by means of the Laplace operator with homogeneous Neumann boundary conditions and its biharmonic counterpart.

An important result of our Green's function research is the fact that it allows to have a fast access to the exact solution of the discrete inpainting problem, in particular when the coefficients of the discrete Green's functions are already known. For this purpose, we have derived several efficient strategies for performing various tasks. In Chapter 6, we will exploit this property to obtain a fast decoding within an image compression application. Moreover, we will see in the next chapters that the presented theory will also lead to practical advantages within the encoding step, as for example for the tonal or spatial optimisation tasks.

It is worth mentioning that the representation of PDE-based inpainting in terms of Green's functions also connects PDE-based image compression to scattered data interpolation with radial basis functions [38]. Many of these basis functions are given as continuous Green's functions on an unbounded domain. In the presented framework we have taken into account the discreteness of digital images and have incorporated image boundaries in a natural way.

Chapter 4

Tonal Optimisation

4.1 Motivation

In PDE-based compression, it is possible to freely choose the mask points and the corresponding tonal values for the inpainting. This enables us to optimise this data in a way such that we obtain the best possible inpainting result. In other words, the reconstruction given by the selected data should be as close as possible to the original image. In particular, instead of relying on the tonal values of the original image, we are free to prescribe modified values at the mask locations. Obviously, this introduces a small local error at the mask points, but hopefully leads to an improved overall approximation.

As mentioned in Section 1.2.2, it would be best to optimise the tonal values in combination with other components such as the mask. However, this chapter mainly focuses on a pure tonal optimisation with a fixed prescribed mask. In the end, it will also be discussed how quantisation can be incorporated into the optimisation process. Based on those results, we will thoroughly discuss the problem of finding a mask in Chapter 5.

Formally, one can describe the tonal optimisation problem for a greyscale image as follows. Colour images can be treated analogously by considering each channel separately. Let $\boldsymbol{c}$ be a fixed mask and $\boldsymbol{L}$ a given linear inpainting operator. Our goal is to find optimal grey values $\boldsymbol{g}$ at the mask positions such that the inpainting result is as close as possible to a prescribed original image $\boldsymbol{f}$. As described in Section 2.3, the function $\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g})$ is defined as the function yielding the inpainting solution for a given mask and some tonal values. The optimised grey-values $\boldsymbol{g}$ are then given as the solution of the least squares problem

$$\underset{\boldsymbol{g} \in \mathbb{R}^{|\Gamma|}}{\arg\min} \, \frac{1}{2} \left\| \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) - \boldsymbol{f} \right\|_2^2 . \tag{4.1}$$

Obviously, the values of $\boldsymbol{g}$ at all non-mask points do not influence the inpainting result and can thus be chosen freely. For the sake of simplicity, we typically set $g_i = 0$ for $i \in \Gamma \backslash K$ and only care about the values at the mask locations K. In the following, different strategies to find a solution of the least-squares problem (4.1) are presented and compared.

The outline of this chapter is as follows. First, we consider an approach relying on inpainting echoes in Section 4.2. A gradient descent strategy for the purpose of tonal optimisation is presented in Section 4.3. Afterwards, two approaches relying on conjugate gradients and primal dual methods are discussed in Sections 4.4 and 4.5. Last but not least, Section 4.5 demonstrates the usefulness of discrete Green's functions to find optimised tonal values. Experiments are conducted in Section 4.7. In the subsequent Sections 4.8 and 4.9, we consider how the tonal optimisation can be performed for a nonlinear inpainting operator or with incorporated quantisation. Finally, the chapter is concluded in Section 4.10.

4.2 Tonal Optimisation with Inpainting Echoes

Historically, this approach has been the first to be proposed for the purpose of tonal optimisation in PDE-based inpainting, and is published in [4]. It bases on the so-called *inpainting echoes* which are defined as follows. Let $\boldsymbol{e}_i$ be the i-th canonical basis vector in $\mathbb{R}^{|\Gamma|}$, i.e. $(\boldsymbol{e}_i)_j = 1$ for $i = j$ and $(\boldsymbol{e}_i)_j = 0$ for $i \neq j$. Then, the inpainting echo $\boldsymbol{b}_i$ for a mask point $i \in K$ is defined as $\boldsymbol{b}_i := \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{e}_i)$. With the help of the inpainting echoes, the solution $\boldsymbol{u}$ of the inpainting problem can be rewritten by employing the linearity of $\boldsymbol{r}$:

$$\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) = \boldsymbol{r}\left(\boldsymbol{c}, \sum_{i \in \Gamma} g_i \boldsymbol{e}_i\right) = \sum_{i \in \Gamma} g_i \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{e}_i) = \sum_{i \in \Gamma} g_i \boldsymbol{b}_i. \tag{4.2}$$

As already mentioned, we are only interested in the values of $\boldsymbol{g}$ at the mask location, and can ignore the vanishing contribution of the non-mask points:

$$\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) = \sum_{i \in K} g_i \boldsymbol{b}_i. \tag{4.3}$$

Writing the inpainting echoes at the mask location into the columns of a matrix $\boldsymbol{B}$, and defining $\boldsymbol{g}_K \in \mathbb{R}^{|K|}$ as the vector $\boldsymbol{g}$ restricted to K, one can write

$$\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) = \boldsymbol{B}\boldsymbol{g}_K, \tag{4.4}$$

which leads to a reformulation of (4.1):

$$\underset{\boldsymbol{g}_K \in \mathbb{R}^{|K|}}{\arg\min} \; \frac{1}{2} \left\| \boldsymbol{B}\boldsymbol{g}_K - \boldsymbol{f} \right\|_2^2 . \tag{4.5}$$

The solution of this least-squares problem is given by solving the corresponding normal equations:

$$\boldsymbol{B}^\top \boldsymbol{B} \boldsymbol{g}_K = \boldsymbol{B}^\top \boldsymbol{f}. \tag{4.6}$$

In [1], it is shown that the matrix $\boldsymbol{B}^\top \boldsymbol{B}$ is invertible. Hence, there exists a unique solution to this problem. To find the solution, one possibility is to solve the linear system of equations (4.6) with direct methods such as LU or QR decomposition [90]. Unfortunately, this is not the best idea as one first has to set up the large system matrix $\boldsymbol{B}^\top \boldsymbol{B}$ of size $|K| \times |K|$. This requires a high computational effort for computing the individual entries given by inner products. Alternatively, iterative solvers have been shown to be much more efficient for solving the normal equations. In [4] for example, we proposed a randomised Gauß–Seidel scheme for this purpose, which works as follows. Starting with the original values of the original image, i.e. $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$, the grey values at the individual mask points are repeatedly updated one after another until convergence. Given the intermediate values $\boldsymbol{g}^k$ at step k, the value at a mask point i is updated via the Gauß–Seidel step

$$g_i^{k+1} = g_i^k + \frac{\boldsymbol{b}_i^\top \left(\boldsymbol{f} - \boldsymbol{u}^k\right)}{\|\boldsymbol{b}_i\|_2^2}, \tag{4.7}$$

with the Euclidean norm of an image $\boldsymbol{a}$ defined as $\|\boldsymbol{a}\|_2 := \sqrt{\langle a, a \rangle}$. As the inpainting echoes are needed in each and every iteration step, it makes sense to precompute them once in the beginning to avoid unnecessary redundant calculations. An attenuation factor ω^k can be added for each iteration to prevent over- or under-shoots during the optimisation. This can be seen as a variant of a *successive over-relaxation scheme* (SOR) [164] with under-relaxation instead of over-relaxation. It makes sense to only activate the attenuation in the beginning, while avoiding it after a few iterations such that it does not slow down the convergence. Thus, a value of $\omega^k = 0.8$ for $k \in \{1, 2\}$, and $\omega^k = 1$ for $k \geq 2$ turns out to give a good performance. The complete algorithm for the tonal optimisation with inpainting echoes is presented in Algorithm 4.1.

Unlike for the discrete Green's functions, there exists no analytic representation of the inpainting echoes. For this reason, the above method suffers from a relatively high computational cost to obtain the individual echoes. Although one can precompute and reuse them for the subsequent iteration steps, they still need to be computed at least once. This is particularly inefficient when a large amount of mask points is present. On an image of size 256×256 for example, the computation of the inpainting echoes for a mask with a density of 4% requires to compute 2621 echoes. This can last several minutes on a modern computer. Moreover, precomputing inpainting echoes leads to high memory requirements. Thus, alternative more efficient tonal optimisation strategies will be discussed in the next Sections.

Despite the aforementioned drawbacks, the inpainting echoes still offer great advan-

Algorithm 4.1. Tonal Optimisation with Inpainting Echoes.

```
Input: Original image $\boldsymbol{f}$, mask $\boldsymbol{c}$, mask indices $K$.
Initialisation: Inpainting $\boldsymbol{u} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})$, grey values $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$, $\varepsilon > 0$.
Compute:
  For all $i \in K$:
      Compute the inpainting echo $\boldsymbol{b}_i = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{e}_i)$.
  Repeat for $k \geq 0$:
      For all $i \in K$:
        1. Compute the correction term $\alpha^k = \frac{\boldsymbol{b}_i^\top (\boldsymbol{f} - \boldsymbol{u}^k)}{|\boldsymbol{b}_i|^2}$.
        2. Update the grey value $g_i^{k+1} = g_i^k + \omega^k \cdot \alpha$.
        3. Update inpainting $\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \omega^k \cdot \alpha \cdot \boldsymbol{b}_i$.
  until the stopping criterion $\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^k\|_2^2 < \varepsilon$ is fulfilled.
Output: Optimised grey values $\boldsymbol{g}$ and corresponding inpainting $\boldsymbol{u}$.
```

tages. For instance, they allow to incorporate a quantisation into the tonal optimisation as we will see in Section 4.9. Moreover, inpainting echoes are very useful when it comes to the optimisation of parameters such as the number of quantisation levels: Once the echoes are computed, all further steps can be performed very fast. Last but not least, it is important to mention that once the inpainting echoes are computed, finding optimal tonal values becomes very easy. The reason for this is that the system matrix $\boldsymbol{B}$ of the associated linear least squares problem has a low complexity. This fact becomes obvious when considering its condition numbers in Table 4.1.

Table 4.1. Condition number of matrix $\boldsymbol{B}$ containing inpainting echoes as columns. The echoes locations are given by different masks with varying density.

mask density	0.01	0.5	1	2	4	8	16
harmonic	10.36	12.11	11.56	10.99	9.37	7.70	5.68
biharmonic	4.11	14.20	14.38	12.91	14.37	11.78	7.87

Obviously, these numbers show that it is not hard to solve the arising linear system of equations once the inpainting echoes are computed. This is a major difference to the condition number of the inpainting matrix itself which we have considered in Section 2.5. Thus, all standard numerical solvers should be able to handle this problem easily.

4.2.1 Examples

We are now in a position to evaluate the benefits of the tonal optimisation. To this end, we consider the example results in Figures 4.1 for homogeneous diffusion inpainting. Examples for biharmonic inpainting are depicted in Figure 4.2. In both cases, the original image *trui* shall be approximated by using different prescribed masks. Besides a random mask and a mask given on a regular grid, also optimised masks for the respective inpainting operators are considered. The latter masks result from probabilistic optimisation strategies [1] which will be discussed in more detail in Chapter 5. All masks contain the same amount of mask points, which is 4% of all pixels. For the given masks, the inpainting results with both the grey values of the original image as well as the optimised grey values are depicted. Note that the focus of this experiment purely lies in the analysis of the tonal optimisation itself, and not in evaluating the influence of the mask or inpainting operator on the reconstructions. This will be the topic of Chapter 5.

As a general observation, it always pays off to perform a tonal optimisation as it helps to tremendously improve the quality of the reconstructions. There is a huge quality gain for all mask configurations and for both inpainting operators. This becomes obvious both from a visual inspection as well as from the error values. One can observe that the error is reduced more the worse the original inpainting is. Nevertheless, even though the inpainting with the original grey values is already very good for the optimised masks, the tonal optimisation is able to further improve the results. This shows that a tonal optimisation is an essential tool for image compression.

4.2.2 Inpainting Echoes vs. Discrete Green's Functions

Similar to the discrete Green's functions, the inpainting echoes can be considered as atoms in a dictionary which allow to assemble the inpainting result. However, there is one major difference: Besides the inpainting operator, boundary conditions and image size, the discrete Green's functions only depend on the sole location of the respective mask point. Thus, they can be considered as the single influence of a mask point on the overall inpainting result. In contrast, the inpainting echoes additionally depend on the complete mask configuration. In particular, the neighbouring mask points have a large impact on the echoes as they represent homogeneous Dirichlet boundary conditions.

To obtain an impression of this effect, Figure 4.3 shows some example inpainting echoes for two different mask configurations. As expected, the inpainting echoes have a peak at the mask point for which they are computed. This mask point will be called the 'main mask point' in the following. Moreover, there is a clear difference between the harmonic and biharmonic inpainting echo. While the harmonic echo fulfils the maximum-minimum principle, the biharmonic echo clearly exceeds the range of [0,1]. Similar to the discrete Green's functions, the harmonic echo shows a visible peak at the centre, while the biharmonic echo reveals a higher smoothness at all locations.

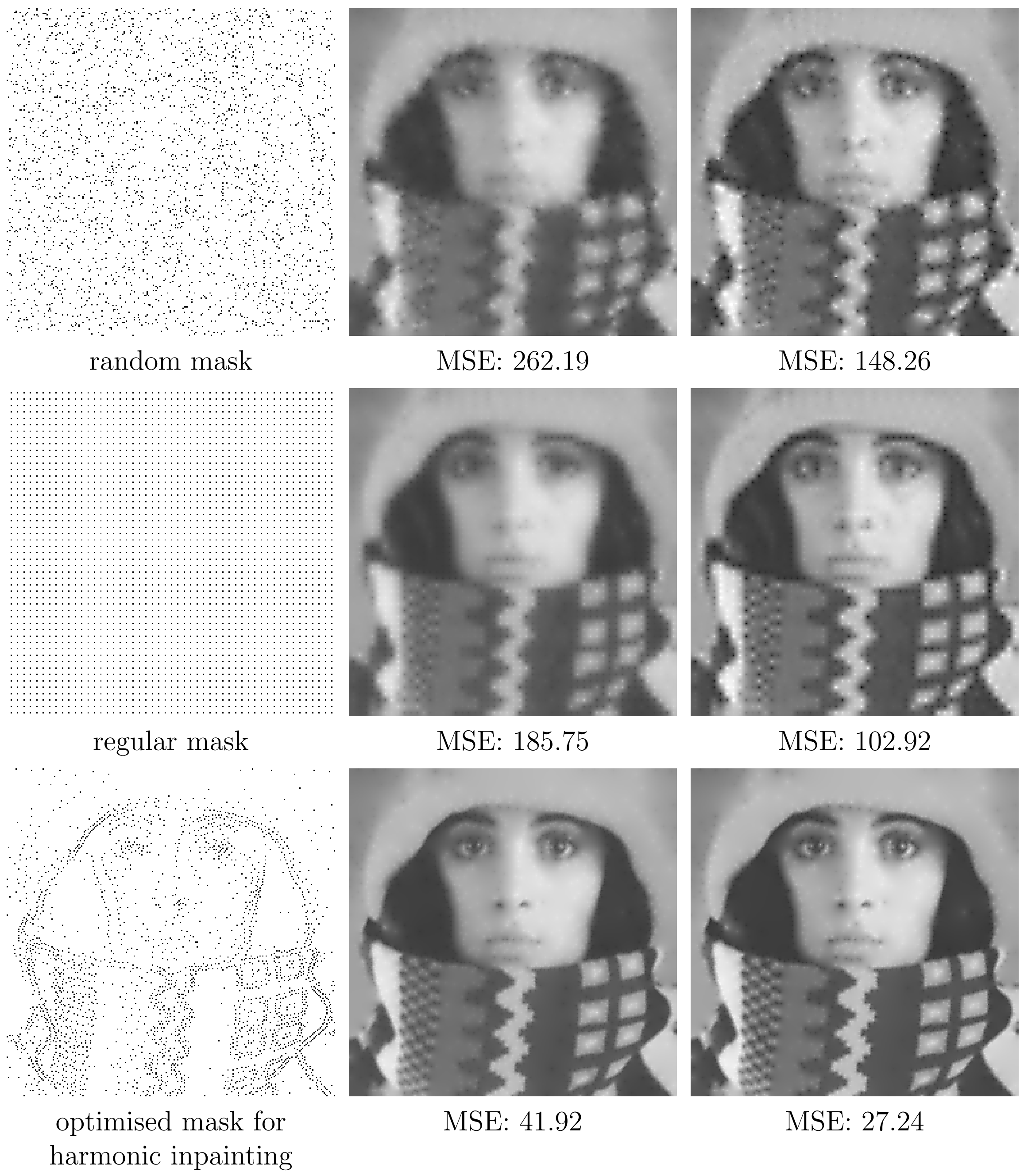

Figure 4.1. Examples of tonal optimisation with harmonic inpainting. **Left column:** Different masks with the same density of 4%. **Centre column:** Inpainting with original grey values as Dirichlet conditions. **Right column:** Inpainting after tonal optimisation.

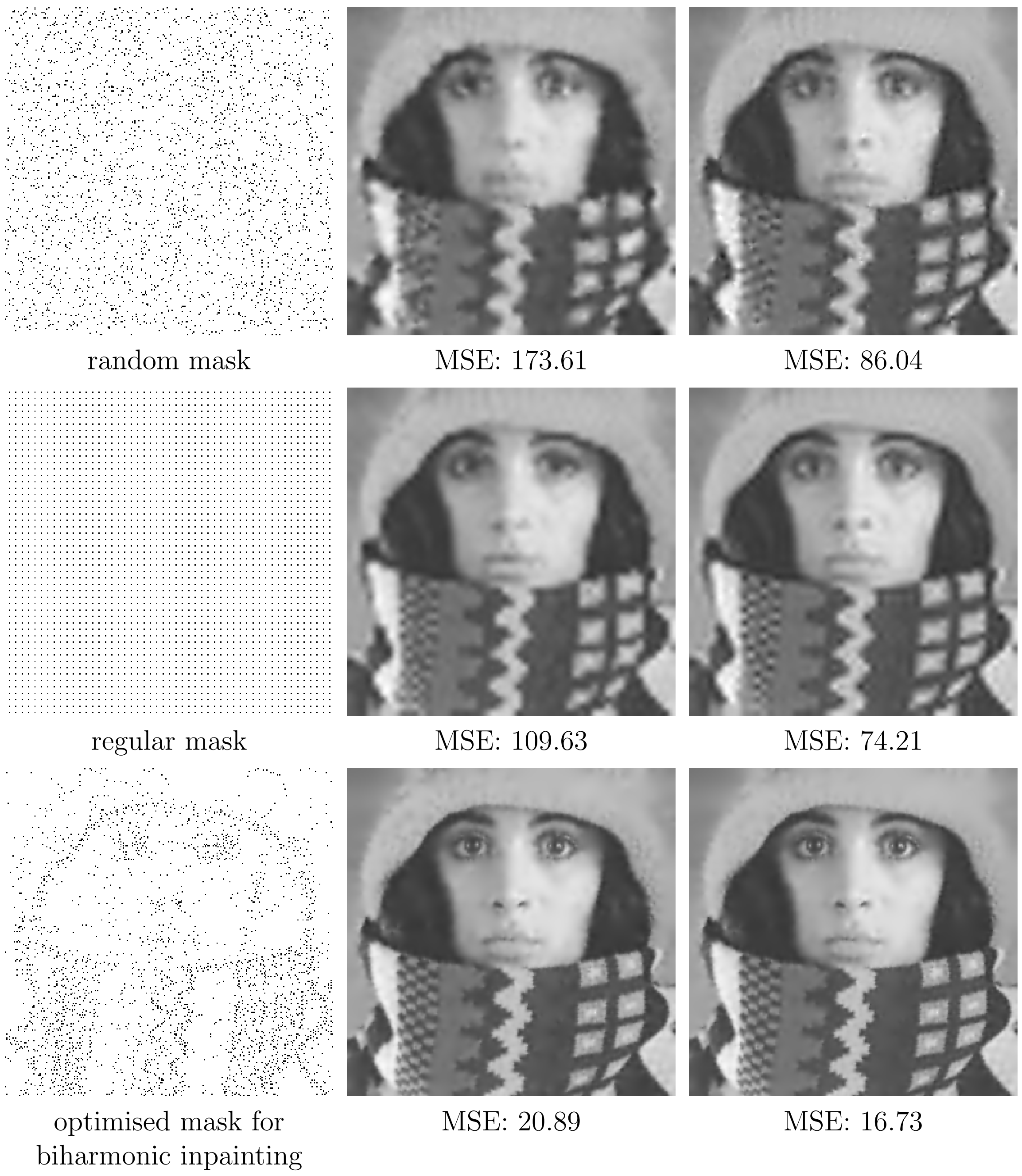

Figure 4.2. Examples of tonal optimisation with biharmonic inpainting. **Left column:** Different masks with the same density of 4%. **Centre column:** Inpainting with original grey values as Dirichlet conditions. **Right column:** Inpainting after tonal optimisation.

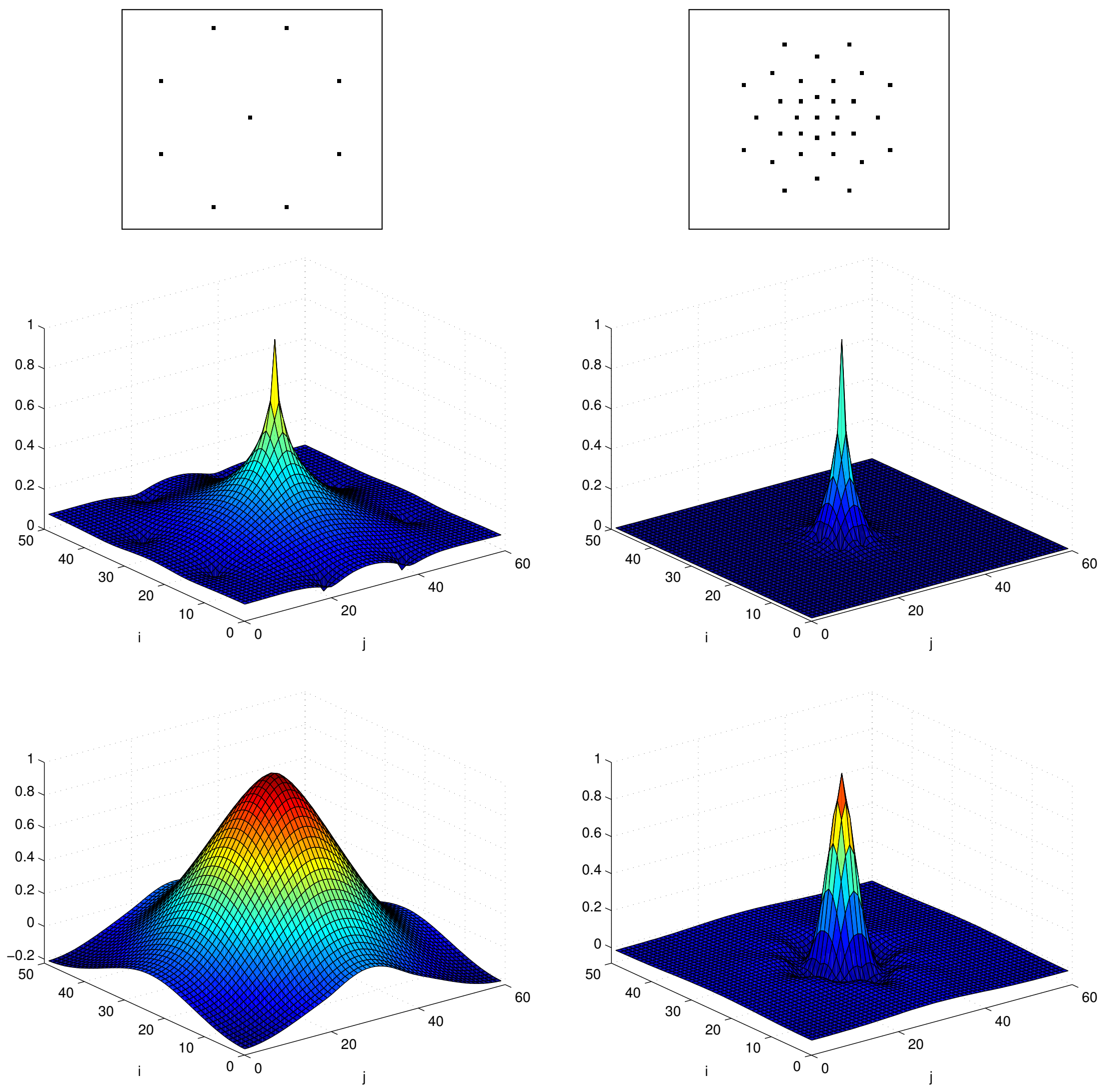

Figure 4.3. Example of inpainting echoes for the mask point (25, 30) in an image of size 50×60. **Top row:** Given masks. **Centre row:** Corresponding echoes with harmonic inpainting. **Right row:** Same with biharmonic inpainting.

For both the harmonic and biharmonic inpainting one can clearly see the influence of the surrounding mask points where the inpainting echoes are equal to zero. They apparently try to tear the echoes down towards zero. The echoes decay more or less depending on the distance of the mask points. For instance, when there is a large gap between the main mask point and the surrounding points, the echoes can be very broad. On the other hand, the decay can also be very rapid when the mask is dense around the main mask point. This shows an interesting property of the inpainting. On the one hand, the mask points have in theory a global influence on the inpainting given by the discrete Green's functions. On the other hand, the actual influence of a grey value on the final inpainting result can be considered very local in practice. The surrounding points of the main mask point seemingly represent an artificial soft boundary of the influence.

4.2.3 Local Approximation of Inpainting Echoes

We have seen in Section 4.2.2 that the influence of a main mask point is restricted by the surrounding mask points. This observation suggests the following strategy to speed up the computation of the inpainting echoes. Instead of computing the echoes at all image pixels, it might be a good idea to restrict them to their respective main region of influence. This avoids the effort to compute the echoes at locations where the main mask point does not have a notable influence at all: The value of the echo can directly be set to zero at those locations.

To identify the support of an inpainting echo, we rely on the well-known Voronoi diagram named after Georgy Voronoi [200]. This diagram allows to find the main region of influence for each mask point by partitioning the image domain into disjoint parts. Thereby, each region is assigned all locations which are closer to the corresponding mask point than to any other mask point. These individual areas are also called Voronoi cells in the literature. An example of such a Vornonoi diagram is depicted in Figure 4.4. Note that the computation of the Voronoi diagram is not very time consuming [1].

Intuitively, an inpainting echo is most dominant within the Voronoi cell of its main mask point. The influence is then reduced in the neighbouring Voronoi cells as the corresponding mask points have a higher influence. By recursively following this reasoning, the influence becomes smaller and smaller the more cells have to be passed. Thus, it seems to be a straightforward idea to restrict the domain for computing an inpainting echo to a certain neighbourhood determined by the Voronoi cells.

It is clear that the restriction of the inpainting echo computation to a smaller region leads to an approximation. If one is interested in the full precision, the echoes have to be computed on the complete image domain. Nevertheless, the hope is to find a good trade-off between quality and performance. This is evaluated in the following experiment. For the original image *trui* and the optimised masks shown in Figures 4.1 and 4.2, a tonal optimisation is performed using approximated inpainting echoes. Both the harmonic as well as the biharmonic inpainting are considered. For a given neighbourhood size d, each individual echo is computed within the region given by joining all Voronoi cells that can be reached without crossing more than d cell boundaries. Thereby, homogeneous Dirichlet boundary conditions are used at all boundaries of the neighbourhood which do not coincide with the actual image boundaries. At the image boundaries, homogeneous Neumann boundary conditions are chosen as usual. Note that the locally approximated inpainting echoes also allow

[1] Implementation of Voronoi diagram courtesy of Dr. Markus Mainberger.

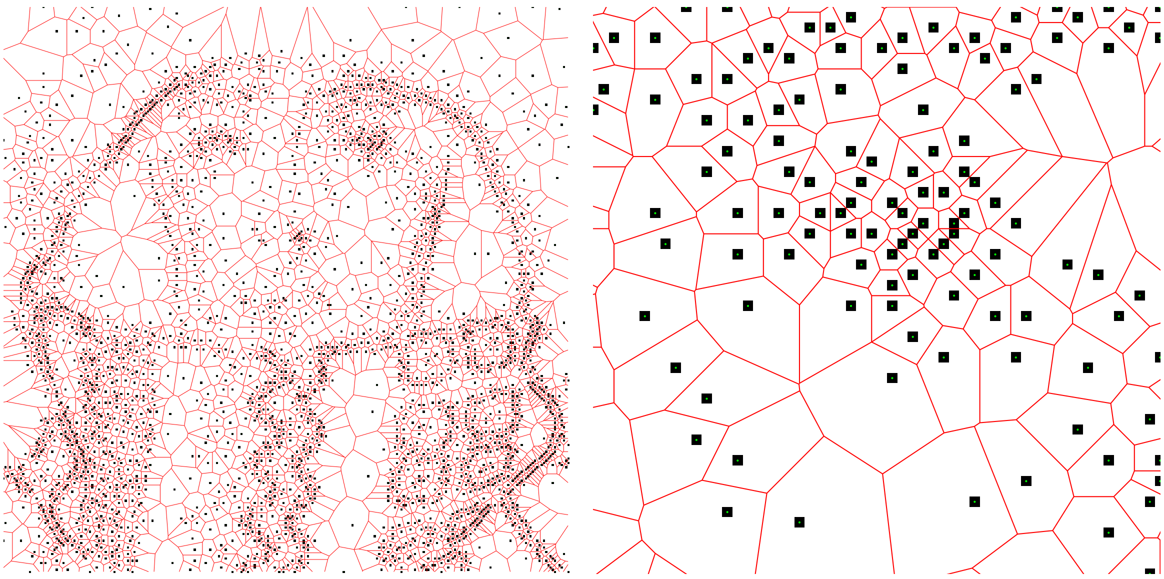
Figure 4.4. Left: Example of Voronoi diagram for optimised mask in Figure 4.1. The red lines show the boundaries of the Voronoi cells, and mask points are depicted in black. **Right:** Zoom-in at the right eye.

to accelerate the actual optimisation of the tonal values in Algorithm 4.1 because only the pixel locations in the individual neighbourhoods have to be evaluated.

The final results are shown in Figure 4.5. Without any local approximation, the computation of the inpainting echoes requires approximately 10 minutes for homogeneous diffusion inpainting, and 29 minutes for biharmonic inpainting. Those reference times and the corresponding final errors are shown as reference. As expected, for smaller neighbourhood sizes d, the quality of the final inpainting decreases, while having a large gain in speed. Conversely, the reconstruction and required time approach more and more the shown original performance for larger values of d. Depending on the desired accuracy, one could for instance use a neighbourhood size of 40 and obtain a good approximation. The MSE values then differ only by 0.07 and 0.13 for harmonic and biharmonic inpainting, respectively. However, this only requires 13% of the original time, which represents an enormous gain. Thus, this strategy is very useful to speed up the approach relying on inpainting echoes when no high accuracy is desired. Nevertheless, we will see in the following that it is possible to obtain both fast and highly accurate results.

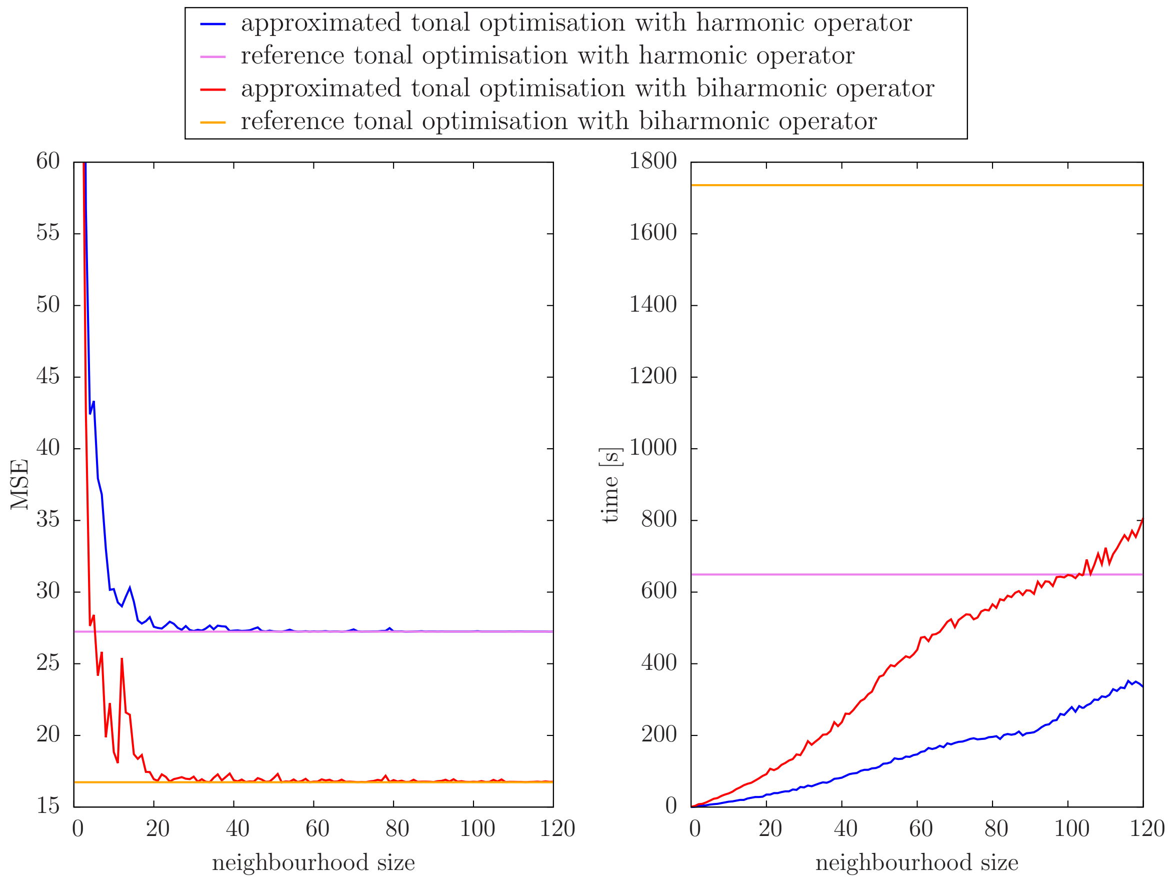

Figure 4.5. Tonal optimisation with locally approximated inpainting echoes based on Voronoi cells. **Left:** Resulting error values for varying neighbourhood sizes. **Right:** Corresponding required times.

4.3 Gradient Descent Approach

In the following approach, we do not want to rely on inpainting echoes to obtain a faster method with smaller memory requirements. To this end, we employ an accelerated gradient descent strategy to find optimal tonal data. We have presented this approach in [1]. For the derivation, let us consider the energy from the original optimisation problem (4.1) which is supposed to be minimised:

$$E(\boldsymbol{g}) = \frac{1}{2} \left\| \boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}\right) - \boldsymbol{f} \right\|_2^2 . \tag{4.8}$$

Starting with an initialisation $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$, a gradient descent scheme minimises the energy E by iteratively updating the current grey values for $k > 0$ as

$$\boldsymbol{g}^{k+1} = \boldsymbol{g}^k - \alpha \, \boldsymbol{\nabla} E(\boldsymbol{g}^k), \tag{4.9}$$

with a step size α and the gradient $\boldsymbol{\nabla} E(\boldsymbol{g}^k)$ depending on the iterates $\boldsymbol{g}^k$. Denoting the inpainting solution at iteration step k by $\boldsymbol{u}^k$, i.e. $\boldsymbol{u}^k := \boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k\right)$, the gradient can be written as

$$\boldsymbol{\nabla} E(\boldsymbol{g}^k) = \boldsymbol{J}^\top \left(\boldsymbol{u}^k - \boldsymbol{f}\right), \tag{4.10}$$

where $\boldsymbol{J}$ is the Jacobian of $\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k\right)$ with respect to the second component. By exploiting the symmetries of the matrices $\boldsymbol{C}$ and $\boldsymbol{L}$, one obtains

$$\begin{aligned} \boldsymbol{J}^\top &= \left((\boldsymbol{C} - (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})^{-1} \boldsymbol{C}\right)^\top \\ &= \boldsymbol{C}\left(\boldsymbol{C} - \boldsymbol{L}(\boldsymbol{I} - \boldsymbol{C})\right)^{-1}. \end{aligned} \tag{4.11}$$

Computing the iterates $\boldsymbol{u}^k$ and the gradient $\boldsymbol{\nabla} E(\boldsymbol{g}^k)$ means to solve a linear system of equations for each of them. This can be done efficiently with a bidirectional multigrid solver as suggested in [125]. As stopping criterion, we consider the relative norm of the gradient, which should approach zero at the optimum. In other words, we stop as soon as

$$\|\boldsymbol{\nabla} E(\boldsymbol{g}^k)\|_2^2 \leq \varepsilon \, \|\boldsymbol{\nabla} E(\boldsymbol{g}^0)\|_2^2, \tag{4.12}$$

with some small number $\varepsilon > 0$.

The main parameter that needs to be specified is the step size α. In the following, two different approaches are presented. First, the classical gradient method with exact line search is explained. Afterwards, we present a novel variant that benefits from an acceleration with a fast explicit diffusion scheme in the sense of Grewenig *et al.* [87].

4.3.1 Exact Line Search

One possibility to select the step size is to optimise it for obtaining the largest possible decay of the energy in each step. This comes down to the least squares problem

$$\underset{\alpha \in \mathbb{R}^+}{\arg\min} \; \frac{1}{2} \left\|\boldsymbol{f} - \boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k - \alpha \boldsymbol{\nabla} E(\boldsymbol{g}^k)\right)\right\|_2^2. \tag{4.13}$$

Exploiting the linearity of $\boldsymbol{r}$ in the second component allows to obtain the minimiser in closed form:

$$\alpha = \frac{(\boldsymbol{f} - \boldsymbol{u}^k)^\top \boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{\nabla} E(\boldsymbol{g}^k)\right)}{\|\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{\nabla} E(\boldsymbol{g}^k)\right)\|_2^2}. \tag{4.14}$$

This strategy is also known as exact line search. It guarantees that the sequence $(\boldsymbol{g}^k)_{k=0,\dots,\infty}$ converges to the minimum of the energy [31]. An algorithmic overview of the classical gradient descent method with exact line search is shown in Algorithm 4.2. It serves as our baseline method.

Algorithm 4.2. Tonal Optimisation with Gradient Descent and Exact Line Search.

```
Input: Original image $\boldsymbol{f}$, pixel mask $\boldsymbol{c}$.
Initialisation: $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$.
Compute:
  Repeat for $k \geq 0$:
    1. Compute the gradient $\boldsymbol{\nabla} E(\boldsymbol{g}^k) = \boldsymbol{J}^\top \left(\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k\right) - \boldsymbol{f}\right)$.
    2. Determine the step size $\alpha$ with Equation (4.14).
    3. Update the tonal data $\boldsymbol{g}^{k+1} = \boldsymbol{g}^k - \alpha\, \boldsymbol{\nabla} E(\boldsymbol{g}^k)$.
  until the stopping criterion (4.12) is fulfilled.
Output: Optimised grey values $\boldsymbol{g}$.
```

4.3.2 Cyclic Acceleration

In order to speed up the gradient descent approach, an accelerated algorithm based on a so-called *fast explicit diffusion (FED)* scheme can be used. First applications of FED to image processing problems go back to Grewenig *et al.* [87]. FED allows to speed up any explicit diffusion-like algorithm that involves a symmetric matrix. While classical explicit schemes employ a constant time step size that has to satisfy a restrictive stability limit, FED schemes involve cycles of time step sizes where up to 50% of them can violate this stability limit. Nevertheless, at the end of each cycle, stability in the Euclidean norm can be guaranteed. In contrast to classical explicit schemes that reach a stopping time of order $O(T)$ in T steps, FED schemes with cycle length T progress to $O(T^2)$. This allows a substantial acceleration.

Since the gradient descent scheme can be seen as an explicit scheme with a symmetric matrix, FED is applicable: If α^* denotes a fixed step size for which (4.9) is stable in the Euclidean norm, one replaces it by the cyclically varying step sizes

$$\alpha_i \;=\; \alpha^* \cdot \frac{1}{2\cos^2\left(\pi \cdot \frac{2i+1}{4T+2}\right)} \quad (i = 0, ..., T-1). \tag{4.15}$$

For large cycle lengths T, one should permute the order of the step sizes to reduce rounding errors; see [208] for more details on this and an exhaustive explanation of the FED framework in general. The FED cycles should be iterated until the stopping criterion (4.12) is fulfilled. A related cyclic optimisation strategy has also been investigated in [177].

In order to determine the individual step sizes within each cycle, we have to find a step size α^* that is within the stability limit. The following result is well-known in optimisation theory [144]: If $E(\boldsymbol{g})$ is continuous and its gradient is Lipschitz continuous, i.e. there is a constant L such that

$$|\boldsymbol{\nabla} E(\boldsymbol{g}_1) - \boldsymbol{\nabla} E(\boldsymbol{g}_2)| \leq L \cdot |\boldsymbol{g}_1 - \boldsymbol{g}_2| \tag{4.16}$$

for all $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, then the gradient descent scheme is stable for all step sizes α^* fulfilling

$$0 < \alpha^* < \frac{2}{L}. \tag{4.17}$$

It is straightforward to verify that in our case L can be chosen as the squared spectral norm of the reconstruction matrix $\boldsymbol{R} := \boldsymbol{M}^{-1}\boldsymbol{C}$, i.e.

$$L = |\boldsymbol{R}|^2 := \rho(\boldsymbol{R}^\top \boldsymbol{R}), \tag{4.18}$$

where $\rho(\boldsymbol{R}^\top \boldsymbol{R})$ denotes the spectral radius of the symmetric matrix $\boldsymbol{R}^\top \boldsymbol{R}$. One possibility to estimate the spectral radius is to use Gershgorin's circle theorem. However, this may give a too pessimistic estimate. Instead, we propose to use the power method to determine the maximum eigenvalue of $\boldsymbol{R}^\top \boldsymbol{R}$, see e.g. [11]. The convergence of this method turns out to be relatively fast, such that one obtains already a reasonable estimate of the spectral radius after 5 iterations.

Although it may appear tempting to choose α^* close to the stability limit $\frac{2}{L}$, this can result in a suboptimal convergence speed, since high frequent error components are damped too slowly. Our experiments suggest that a good choice for α^* is two third of the stability limit:

$$\alpha^* = \frac{4}{3L}. \tag{4.19}$$

Similar strategies are also common e.g. in multigrid approaches that use a damped Jacobi method with damping factor $\frac{2}{3}$ as a baseline solver [35].

An overview over all steps to perform tonal optimisation with FED-accelerated gradient descent is given in Algorithm 4.3. Considering the cycle length, one faces the following trade-off. If T is chosen too small, the acceleration given by the larger time step sizes is restricted as they cannot be too large. On the other hand, it is also not good to choose a very large T, as one always has to perform all inner cycle steps within each outer iteration step. Thus, it can happen that even though one has almost reached a solution fulfilling the stopping criterion, another full cycle is performed. This can lead to a much higher precision and a consequently higher runtime.

To get an impression of realistic runtimes of both gradient descent algorithms for tonal optimisation, let us consider the following experiment. We consider the inpainting problem with the original image trui along with the optimised mask from Figure 4.1 having a density of 4%. The tonal optimisation shall be performed for homogeneous diffusion inpainting. The stopping parameter is set to $\varepsilon := 0.001$ for all

Algorithm 4.3. Tonal Optimisation with Gradient Descent and Cyclic Acceleration.

```
Input: Original image $\boldsymbol{f}$, pixel mask $\boldsymbol{c}$, FED cycle length $T$.
Initialisation:
    1. Set $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$.
    2. Estimate $L$ in (4.18) with the power method.
    3. Determine the FED time steps $\alpha_0, \dots, \alpha_{T-1}$ according to (4.15) with $\alpha^* = \frac{4}{3L}$.
       If necessary, permute them.
Compute:
  Repeat for $k \geq 0$:
    1. $\boldsymbol{g}^{k,0} := \boldsymbol{g}^k$
    2. For $i = 0, \dots, T-1$ do
        i. Compute the gradient $\boldsymbol{\nabla} E(\boldsymbol{g}^{k,i}) = \boldsymbol{J}^\top \left(\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^{k,i}\right) - \boldsymbol{f}\right)$.
        ii. Update the tonal data: $\boldsymbol{g}^{k,i+1} = \boldsymbol{g}^{k,i} - \alpha_i \, \boldsymbol{\nabla} E(\boldsymbol{g}^{k,i})$.
    3. Set $\boldsymbol{g}^{k+1} = \boldsymbol{g}^{k,T}$.
  until the stopping criterion (4.12) is fulfilled.
Output: Optimised grey values $\boldsymbol{g}$.
```

our experiments. As a result, the exact line search algorithm requires 458 seconds to perform 262 iterations. A corresponding FED accelerated algorithm with $\alpha^* = 0.01$ needs only 77 seconds to compute 4 cycles of length $M = 15$. This illustrates the favourable performance of the FED-accelerated gradient descent method.

Since the FED algorithm is based on an explicit scheme, it is also well-suited for implementations on parallel hardware such as GPUs. Unfortunately, the multigrid solver which is used for computing an inpainting solution or to compute the Jacobian in (4.11) in every iteration step is not perfectly suited for a GPU implementation: Especially on coarser scales, only few pixels are left such that the full capacity of all processing units on a GPU might not be exploited. Instead, it makes sense to rely on an FED algorithm here which is perfectly suited for the GPU. Altogether, the GPU variant can lead to very substantial additional accelerations compared to the CPU implementation. For instance, the tonal optimisation example above only requires 6 seconds on a GPU, which is approximately 13 times faster compared to the 77 seconds on a CPU. Further evaluations of the performance will will be performed in Section 4.7.

4.4 Conjugate Gradient Approach

Instead of performing a gradient descent of the energy (4.1), it is also possible to find the minimiser as solution of the associated normal equations. For the tonal optimisation problem, they read as

$$\boldsymbol{R}^\top \boldsymbol{R}\boldsymbol{g} = \boldsymbol{R}^\top \boldsymbol{f}, \tag{4.20}$$

with $\boldsymbol{R} := \boldsymbol{M}^{-1}\boldsymbol{C}$. Setting up the system matrix $\boldsymbol{R}^\top \boldsymbol{R}$ would again involve computing the inpainting echoes. As discussed earlier, this is not efficient. A better idea is to rely on a numerical solver which uses an iterative scheme to find the solution. The advantage of such a strategy is that it does not require to set up or even invert the matrix explicitly. The gradient descent strategy in Section 4.3 can be interpreted as one specific example in this context. In particular, we have seen that subsequently applying the matrices $\boldsymbol{R}$ and $\boldsymbol{R}^\top$ can be efficiently done.

In the literature, there exist many alternative approaches, see [164, 165] for example. Among them, the conjugate gradient strategy turns out to be a promising technique as it optimises the search direction in every iteration. In contrast to a pure gradient descent strategy, the current residuum as well as the previous search directions influence the search strategy in each step. Since the system matrix is positive semi-definite and symmetric, the conjugate gradient approach can be applied. For more details on the algorithm, please see [164]. Unfortunately, the inner products within the optimisation procedure complicate parallel processing of the individual pixels. Thus, a GPU implementation is not as easy to realise as the gradient descent scheme.

4.5 Primal-Dual Algorithm

Primal-dual approaches have become more and more popular in recent years for solving minimisation problems in the context of image processing [42]. They rely on performing a gradient descent on the primal variable, and at the same time a gradient ascent on the dual variable. The promising performance of those primal-dual approaches inspired Hoeltgen and Weickert to use them also for the purpose of tonal optimisation with homogeneous diffusion inpainting [93]. The proposed method showed a good performance and a tremendous speed-up compared to the method relying on inpainting echoes.

In the following, we revisit the method presented in [93], and incorporate several improvements and extensions. In contrast to the original method, the introduction of an extra variable is avoided which simplifies the algorithm and reduces the memory requirements. Moreover, a diagonal preconditioning of the system matrix is employed as proposed in [156]. This drastically improves the convergence of the algorithm.

While the focus in [93] is on the harmonic inpainting operator, we consider a general linear inpainting operator $\boldsymbol{L}$ in the following. This also allows to perform a tonal optimisation for the biharmonic operator for example. Moreover, besides a CPU implementation, the performance of the method will also be evaluated on a GPU as the algorithm is perfectly suited for such a parallel hardware. Some of the performance results of this GPU implementation have already been presented in [93].

Let us reconsider the minimisation problem (4.1). By defining the inpainting solution $\boldsymbol{u} := \boldsymbol{r}\,(\boldsymbol{c}, \boldsymbol{g}) = \boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{g}$ for some grey values $\boldsymbol{g}$, one can rewrite the objective:

$$\underset{\boldsymbol{u},\boldsymbol{g}\in\mathbb{R}^{|\Gamma|}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{u}-\boldsymbol{f}\|_2^2 \qquad \text{subject to} \qquad \boldsymbol{u} = \boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{g}. \tag{4.21}$$

Due to the fact that the mask is binary, the inpainting solution $\boldsymbol{u}$ is identical to the prescribed tonal values $\boldsymbol{g}$ at the mask locations, i.e. $\boldsymbol{C}\boldsymbol{g} = \boldsymbol{C}\boldsymbol{u}$, because it is an interpolation of the prescribed data. For this reason, one can decrease the number of unknowns, and the constraint becomes:

$$\begin{aligned} \boldsymbol{u} = \boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{g} \qquad &\Leftrightarrow \qquad \boldsymbol{u} = \boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{u} && (4.22)\\ &\Leftrightarrow \qquad \boldsymbol{M}\boldsymbol{u} = \boldsymbol{C}\boldsymbol{u} && (4.23)\\ &\Leftrightarrow \qquad \boldsymbol{R}\boldsymbol{u} = \boldsymbol{0}, && (4.24) \end{aligned}$$

with $\boldsymbol{R} := \boldsymbol{M} - \boldsymbol{C} = -(\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L}$. By defining the indicator function $\iota_{\boldsymbol{0}}$ as

$$\iota_{\boldsymbol{a}}(\boldsymbol{x}) := \begin{cases} 0 & \text{if } \boldsymbol{x} = \boldsymbol{a}, \\ \infty & \text{if } \boldsymbol{x} \neq \boldsymbol{a}, \end{cases} \tag{4.25}$$

we can reformulate Equation (4.21) to:

$$\underset{\boldsymbol{u}\in\mathbb{R}^{|\Gamma|}}{\arg\min} \left\{ \frac{1}{2}\|\boldsymbol{u}-\boldsymbol{f}\|_2^2 + \iota_{\boldsymbol{0}}(\boldsymbol{R}\boldsymbol{u}) \right\}. \tag{4.26}$$

The advantage of this minimisation problem is that it does not involve the inverse of the inpainting matrix anymore. To find a solution and hence the tonal optimised grey values at the mask locations, the primal-dual method from [42] can be used. It iteratively updates the primal and dual variables, and only incorporates matrix vector multiplications as computationally most demanding operations. As already mentioned, we additionally use the diagonal preconditioning proposed in [156] to improve the convergence. This is achieved by adapting the step sizes $\boldsymbol{T} := \operatorname{diag}(\boldsymbol{\tau})$ and $\boldsymbol{\Sigma} := \operatorname{diag}(\boldsymbol{\sigma})$ to the pixel locations and to the operator $\boldsymbol{R}$. As suggested in [156], the step sizes can be computed by collecting the rows and columns of $\boldsymbol{R}$, respectively:

$$\tau_j = \frac{1}{\sum_{i=0}^{|\Gamma|} |R_{i,j}|^{2-\alpha}}, \qquad \sigma_i = \frac{1}{\sum_{j=0}^{|\Gamma|} |R_{i,j}|^{\alpha}}. \tag{4.27}$$

Algorithm 4.4. Tonal Optimisation with Prima-Dual Algorithm.

```
Input: Original image $\boldsymbol{f}$, pixel mask $\boldsymbol{c}$.
Initialisation:
    1. Calculate precondition matrices $\boldsymbol{T}$ and $\boldsymbol{\Sigma}$ given by (4.27).
    2. Set $\boldsymbol{u}^0 = \boldsymbol{f}$, $\widehat{\boldsymbol{u}}^0 = \boldsymbol{f}$, $\boldsymbol{y}^0 = \boldsymbol{0}$, $theta \in [0,1]$, $\varepsilon > 0$.
Compute:
  Repeat for $k \geq 0$:
    1. $\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \boldsymbol{\Sigma}\boldsymbol{R}\,\widehat{\boldsymbol{u}}^k$.
    2. $\boldsymbol{u}^{k+1} = (\boldsymbol{I}+\boldsymbol{T})^{-1}(\boldsymbol{u}^k - \boldsymbol{T}(\boldsymbol{R}^\top \boldsymbol{y}^{k+1} - \boldsymbol{f})$.
    3. $\widehat{\boldsymbol{u}}^{k+1} = \widehat{u}^{k+1} + \theta(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k)$.
  until stopping criterion $\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^k\|_2^2 < \varepsilon$ is fulfilled.
Output: Inpainting $\boldsymbol{u}$ containing the optimised grey values at the mask locations.
```

We choose the parameter $\alpha \in [0, 2]$ to be 1 as this yields good results. An overview of the primal-dual solver is presented in Algorithm 4.4. The parameter θ steers the amount of over-relaxation and is usually fixed to 1. Note that the computation of the inverse of $\boldsymbol{I}+\boldsymbol{T}$, with $\boldsymbol{I}$ being the unit matrix, is straightforward as it is a diagonal matrix.

Besides a primal-dual solver, also different strategies to find a minimiser of (4.21) have been employed in the literature. In [51] for example, the authors rely on the memory-limited version of the Broyden-Fletcher-Goldfarb-Shanno algorithm (L-BFGS), which is an approximation of the Newton's method. Another possibility is the LSQR method [149], which is used in [93] for the purpose of performing a tonal optimisation. However, we will stick to the primal-dual strategy as it is easy to implement and hence no additional sophisticated mathematical library is required. Moreover, it is very well suited for parallel computations on a GPU.

4.6 Tonal Optimisation with Discrete Green's Functions

We know from Section 3.4 that the inpainting result for a specific mask is given as superposition of the discrete Green's functions at those locations. A natural question is if we can use this knowledge for the tonal optimisation problem. Luckily, we are able to do so by transferring the problem of finding tonal data to the equivalent problem of finding optimal coefficients of the corresponding discrete Green's functions. This idea is elaborated in the following.

We assume a fixed mask $K \in \Gamma$ to be given with $L := |K|$ number of points. Moreover, we define $\boldsymbol{H} \in \mathbb{R}^{|\Gamma| \times (L+1)}$ as the matrix containing the discrete Green's functions at all mask locations in its columns, with an additional column $\mathbf{1} \in \mathbb{R}^{|\Gamma|}$ consisting of 1's in all entries for the constant value:

$$\boldsymbol{H} := \left(\boldsymbol{g}_{m_1}^0, \dots, \boldsymbol{g}_{m_L}^0, \mathbf{1}\right). \tag{4.28}$$

As in Section 3.4, $m_0, \dots, m_L$ represent the indices of the mask points. Additionally, let $\boldsymbol{x} := (a_1, \dots, a_L, c)^\top \in \mathbb{R}^{L+1}$ be the vector consisting of the coefficients of the individual Green's functions as well as of the constant c. Note that any inpainting solution $\boldsymbol{u}$ can be expressed as $\boldsymbol{Hx}$, since the Green's functions are a generating system of all inpainting solutions for this specific mask, cf. Section 3.4. We further know from the solvability condition (3.53) that

$$\sum_{k=1}^{L} a_{m_k} = 0 \qquad \Longleftrightarrow \qquad a_L = -\sum_{k=1}^{L-1} a_k. \tag{4.29}$$

By defining

$$\widetilde{\boldsymbol{H}} := \left(\boldsymbol{g}_{m_1}^0 - \boldsymbol{g}_{m_L}^0, \dots, \boldsymbol{g}_{m_{L-1}}^0 - \boldsymbol{g}_{m_L}^0, \mathbf{1}\right) \in \mathbb{R}^{|\Gamma| \times L}, \tag{4.30}$$

and accordingly

$$\widetilde{\boldsymbol{x}} := (a_{m_1}, \dots, a_{m_{L-1}}, c)^\top \in \mathbb{R}^L, \tag{4.31}$$

we obtain that $\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}$ is a valid inpainting solution for all $\widetilde{\boldsymbol{x}} \in \mathbb{R}^L$. Thus, we propose to perform a tonal optimisation by finding the optimal coefficients as

$$\underset{\widetilde{\boldsymbol{x}} \in \mathbb{R}^L}{\arg\min}\ \frac{1}{2} \left\| \widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}} - \boldsymbol{f} \right\|_2^2. \tag{4.32}$$

It is clear that this linear least squares problem has a unique solution as it represents a minimisation of a strictly convex energy which is bounded form below. Theorem 4.1 shows that this novel strategy in terms of discrete Green's functions is in fact equivalent to the standard tonal optimisation formulation (4.1).

Theorem 4.1 (Equivalence of Tonal Optimisation Strategies)**.** *Let $\boldsymbol{g}^*$ be the minimiser of* (4.1)*, and $\widetilde{\boldsymbol{x}}^*$ the minimiser of* (4.32)*. Then the resulting tonal optimised inpainting solutions are identical, i.e.*

$$\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}^*) = \widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}^*. \tag{4.33}$$

Proof. From Section 3.4 we know that for the inpainting result $\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}^*)$ we can find a vector $\widetilde{\boldsymbol{x}}$ such that $\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^{*}\right) = \widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}$. Thus, we have

$$\frac{1}{2}\left\|\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^{*}\right) - \boldsymbol{f}\right\|_2^2 \;\geq\; \frac{1}{2}\left\|\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}^{*} - \boldsymbol{f}\right\|_2^2 . \tag{4.34}$$

On the other hand, $\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}^{*}$ represent an inpainting result which consists of some tonal values $\boldsymbol{g}$ at the mask points, and therefore

$$\frac{1}{2}\left\|\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^{*}\right) - \boldsymbol{f}\right\|_2^2 \;\leq\; \frac{1}{2}\left\|\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}^{*} - \boldsymbol{f}\right\|_2^2 . \tag{4.35}$$

Hence, the energies for both minimisers are the same. We know that both linear least squares problems have a unique solution. Thus, the one-to-one correspondence between the inpainting solutions tells us that they are identical. This proofs the claim. □

It is known that the minimiser of the energy in (4.32) is given as solution of the normal equations

$$\widetilde{\boldsymbol{H}}^{\top}\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{x}}^{*} = \widetilde{\boldsymbol{H}}^{\top}\boldsymbol{f}, \tag{4.36}$$

see for example [165]. The big advantage of this new approach is that the theory on discrete Green's functions allows to efficiently set up the system matrix $\widetilde{\boldsymbol{H}}^{\top}\widetilde{\boldsymbol{H}} \in \mathbb{R}^{L\times L}$ directly. More specifically, the entries are given for $i, j = 1, \ldots, L-1$ as

$$\begin{aligned}\left(\widetilde{\boldsymbol{H}}^{\top}\widetilde{\boldsymbol{H}}\right)_{i,j} &= \left\langle \boldsymbol{g}^0_{m_i} - \boldsymbol{g}^0_{m_L}, \boldsymbol{g}^0_{m_j} - \boldsymbol{g}^0_{m_L}\right\rangle \\ &= \left\langle \boldsymbol{g}^0_{m_i}, \boldsymbol{g}^0_{m_j}\right\rangle - \left\langle \boldsymbol{g}^0_{m_i}, \boldsymbol{g}^0_{m_L}\right\rangle - \left\langle \boldsymbol{g}^0_{m_j}, \boldsymbol{g}^0_{m_L}\right\rangle + \left\langle \boldsymbol{g}^0_{m_L}, \boldsymbol{g}^0_{m_L}\right\rangle .\end{aligned} \tag{4.37}$$

As we have seen in Section 3.3.4 and Section 3.3.5, we are able to compute the Green's functions as well as the inner products in a very efficient way. Note that all inner products with $\boldsymbol{g}^0_{m_L}$ can be precomputed to further speed up the computations. Moreover, we can exploit the symmetry of the matrix, which means that the lower triangular part can be obtained directly after having computed the upper triangular part. In a similar fashion, the right hand side of the normal equations can be constructed with the help of the fast inner products.

Due to the the fact that all canonical Green's functions have zero mean, the last row and last column of $\widetilde{\boldsymbol{H}}^{\top}\widetilde{\boldsymbol{H}}$ only consist of zeros except for the entry at $i = j = L$ which is equal to $|\Gamma|$. Thus, we directly obtain the constant c:

$$c = \frac{1}{|\Gamma|}\left\langle \boldsymbol{f}, \boldsymbol{1}\right\rangle . \tag{4.38}$$

We know from Section 3.4 that c represents the mean value of the tonal optimised inpainting solution. Thus, (4.38) shows that the optimal mean of the inpainting is identical to the mean value of the original image. This is a nice new observation which results from interpreting the tonal optimisation problem in terms of the discrete Green's functions. For finding the remaining coefficients as solution of the normal equations, we employ a very efficient numerical solver based on a Cholesky factorisation, which is particularly well suited as it exploits the positive definiteness of the system matrix. For more details, see [90]. The proof of this property is given in Theorem 4.2.

Theorem 4.2 (Positive Definiteness of System Matrix)**.** *The matrix* $\widetilde{\boldsymbol{H}}^\top \widetilde{\boldsymbol{H}}$ *is positive definite.*

Proof. We have to show that $\boldsymbol{x}^\top \widetilde{\boldsymbol{H}}^\top \widetilde{\boldsymbol{H}} \boldsymbol{x} > 0$ for all $\boldsymbol{x} \in \mathbb{R}^L \backslash \{\boldsymbol{0}\}$. It is easy to see that $\boldsymbol{x}^\top \widetilde{\boldsymbol{H}}^\top \widetilde{\boldsymbol{H}} \boldsymbol{x} = \|\widetilde{\boldsymbol{H}} \boldsymbol{x}\|_2^2 \geq 0$ and hence the positive semi-definiteness of the matrix follows directly. To show that 0 is excluded, we employ the solvability condition and obtain:

$$\widetilde{\boldsymbol{H}} \boldsymbol{x} = \sum_{k=1}^{L-1} \left[a_k \left(\boldsymbol{g}_{m_k}^0 - \boldsymbol{g}_{m_L}^0 \right) \right] + c \cdot \boldsymbol{1} = \sum_{k=1}^{L} a_k \, \boldsymbol{g}_{m_k}^0 + c \boldsymbol{1} \,. \tag{4.39}$$

With the help of the representation of the discrete Green's functions from Theorem 3.10, we can rewrite this expression:

$$\begin{aligned} \widetilde{\boldsymbol{H}} \boldsymbol{x} &= \sum_{k=1}^{L} a_k \sum_{\substack{m=0 \\ (m,n)\neq(0,0)}}^{M-1} \sum_{n=0}^{N-1} \left[\frac{1}{\lambda_{m,n}} (\boldsymbol{g}_{m,n})_{m_k} \cdot \boldsymbol{g}_{m,n} \right] + c \cdot \boldsymbol{1} \\ &= \sum_{\substack{m=0 \\ (m,n)\neq(0,0)}}^{M-1} \sum_{n=0}^{N-1} \left[\left(\frac{1}{\lambda_{m,n}} \sum_{k=1}^{L} a_k \cdot (\boldsymbol{g}_{m,n})_{m_k} \right) \boldsymbol{g}_{m,n} \right] + \sqrt{|\Gamma|} \cdot c \cdot \boldsymbol{g}_{0,0} \,. \end{aligned} \tag{4.40}$$

We know that the $\boldsymbol{g}_{m,n}$ form an orthogonal set of eigenvectors and that the eigenvalues $\lambda_{m,n}$ are non-zero for all $(m,n) \neq (0,0)$. Thus, in order for (4.40) to be the zero vector, we must have $c = 0$ as well as for all m, n:

$$0 = \frac{1}{\lambda_{m,n}} \sum_{k=1}^{L} a_k \cdot (\boldsymbol{g}_{m,n})_{m_k} = \frac{1}{\lambda_{m,n}} \sum_{k=1}^{L} a_k \cdot (\boldsymbol{g}_{m_k})_{m,n} \,. \tag{4.41}$$

Again, we know that the $\boldsymbol{g}_{m_k}$ are orthogonal, and consequently, all entries of $\widetilde{\boldsymbol{x}}$ have to be zero. Thus, there is no non-trivial vector $\widetilde{\boldsymbol{x}}$ such that $\|\widetilde{\boldsymbol{H}} \boldsymbol{x}\|_2^2 = 0$, which shows the positive definiteness. □

An overview over all steps to perform a tonal optimisation is given in Algorithm 4.5.

Algorithm 4.5. Tonal Optimisation with Discrete Green's Functions.

Input: Original image $\boldsymbol{f}$, mask indices K.

Compute:

1. Obtain the Green's mother function as described in Section 3.3.5.
2. Calculate the Green's functions $\boldsymbol{g}^0_{k,\ell}$ for all $(k,\ell) \in K$ from the Green's mother function with Algorithm 3.3.
3. Set up the system matrix $\widetilde{\boldsymbol{H}}^\top \widetilde{\boldsymbol{H}}$ and $\widetilde{\boldsymbol{H}}^\top \boldsymbol{f}$ with fast inner products, see Section 3.3.4.
4. Solve (4.36) with the help of a Cholesky factorisation to find optimal coefficients $a_1, .., a_{L-1}$ and c.
5. Compute a_L with solvability condition, see Equation (4.29).
6. Obtain the inpainting $\boldsymbol{u}$ with Algorithm 3.1 and subsequently adding c.

Output: Inpainting solution $\boldsymbol{u}$ with optimised grey values at the mask locations.

Compared to other methods for the tonal optimisation which use the standard formulation (4.1), this new approach has several advantages. First of all, the new method does not entail any parameters which need to be chosen. In other approaches, one typically has to choose appropriate stopping criteria for the numerical solvers of the inpainting process as well as for the tonal optimisation. This can be very cumbersome. Moreover, the iterative solvers of other approaches only lead to approximate solutions. With the help of the Green's functions, we directly arrive at the exact solution up to machine precision, because the problem is solved analytically after setting up the normal equations directly. Last but not least, it is straightforward to extend this approach to higher powers of the Laplacian as inpainting operator by adapting the powers of the eigenvalues. In contrast, all other approaches require more sophisticated adaptations for this purpose or even a redesign of the underlying numerical algorithm.

As for the inpainting echoes, one can also have a look at the condition number of the system matrix $\widetilde{\boldsymbol{H}}$ arising in the approach using the discrete Green's functions. The results are shown in Table 4.2.

Table 4.2. Condition number of matrix $\widetilde{\boldsymbol{H}}$ containing discrete Green's functions for the Laplacian as columns. The echoes locations are given by different masks with varying density.

mask density	0.01	0.5	1	2	4	8	16
harmonic	11	627	1207	2558	5872	10158	20491
biharmonic	127	1035127	4310728	19169138	117196309	300948395	944357153

Compared to the condition numbers in Table 4.1 with the inpainting echoes, the discrete Green's functions are apparently more linear dependent, hence leading to a harder problem. This becomes more obvious in particular for higher mask densities and higher orders of the operator. Nevertheless, the Cholesky factorisation is able to cope with this issue in practical scenarios.

4.7 Runtime Comparison

In the following, the performance of the individual tonal optimisation approaches is evaluated. Again, the image trui is used as original image, and the same masks based on the approach of Belhachmi *et al.* with densities varying from 0.01% to 16% are used. The tonal optimisation is performed for the harmonic as well as the biharmonic operator. First, we will consider a pure CPU implementation. Afterwards, the performance of some tonal optimisation strategies implemented on a GPU hardware is compared. Note that the grey value optimisation problem is strictly convex which means that all convergent algorithms yield the same minimiser and are therefore qualitatively equivalent. Thus, they are graded only by their respective runtimes.

As discussed in Section 4.2, the approach relying on inpainting echoes already requires a lot of time due to the computation of the inpainting echoes. Even relying on a parallel computation of their local approximation using the Voronoi diagram is still too slow and suffers from low precise results. For these reasons, we will focus on the remaining more efficient methods, namely the cyclic accelerated gradient descent, the conjugate gradient approach, the primal-dual approach as well as the tonal optimisation relying on discrete Green's functions. Whenever an algorithm runs for more than one hour the program is aborted and the result is not considered, because such a long runtime is far from being acceptable.

4.7.1 CPU Performance

Similar to Section 3.5, the experiments are performed with C implementations on a desktop CPU with Intel Xeon processor (3.2GHz). All results are depicted in Figure 4.6. Note that the time axis is plotted on a logarithmic scale for a better visibility.

Considering the approach with Green's functions, one observes that the algorithm almost needs no time for smaller mask densities. For higher densities, the time increase approximately linearly. An important aspect is that the runtime is independent of the employed inpainting operator. Moreover, note that the result is highly accurate up to machine precision as the tonal values are optimised analytically. All other methods rely on iterative solvers which are only able to approximate the solution. Nevertheless, the Green's approach outperforms all other methods for masks with densities of up to 4%. This demonstrates the advantage of expressing the inpainting solution in terms of those atomic components. The reason clearly lies in the fact that the complexity of the method decreases with a smaller number of mask points.

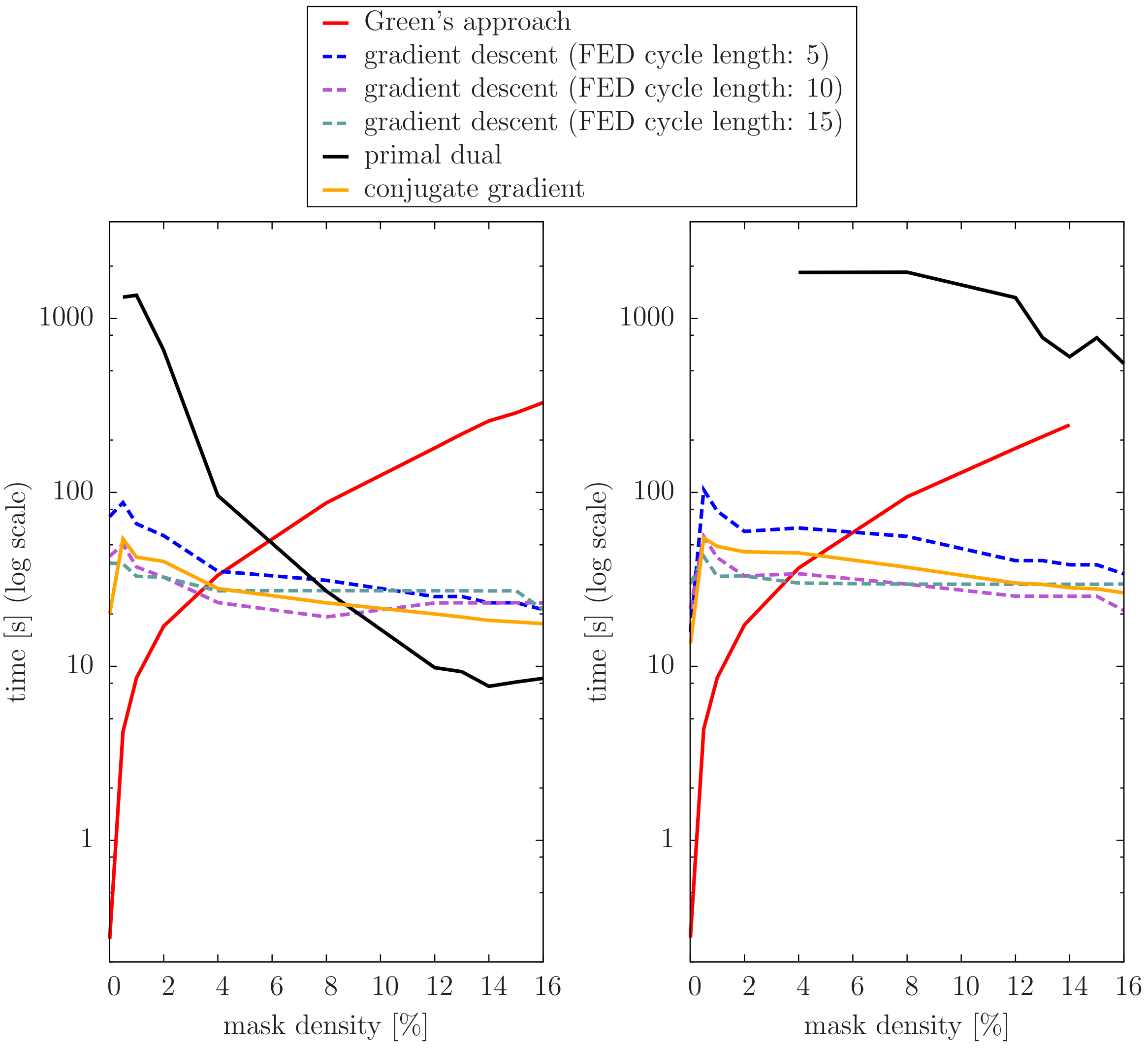

Figure 4.6. Runtime comparison of different tonal optimisation strategies for various mask densities on a CPU. **Left:** Harmonic inpainting operator. **Right:** Biharmonic inpainting operator.

Similar to the pure inpainting, this is converse to the original inpainting formulation which is more efficient for higher mask densities. Additionally, one can observe that the runtime is independent of the inpainting operator. In particular, one can even use an arbitrary power of the Laplacian with the same runtime.

The analytic approach only has problems for the biharmonic operator and very large mask densities where numerical inaccuracies occur even with double precision which prevents to compute a solution. As seen in Table 4.2, the condition numbers of the system matrix can become very large. This means that smallest eigenvalue becomes closer and closer to zero. The Cholesky factorisation is not able to handle this numerically at some point as it relies on the positive definiteness of the matrix.

The gradient descent approach seems to have a comparably more stable runtime over all tested mask densities, even though one can recognise an improved performance the more mask points are present. Furthermore, one can observe a slightly higher runtime for the biharmonic operator compared to the harmonic one. Considering the FED cycle length, it seems that a value of 10 gives good performance. For some densities, a cycle length of 15 can be slightly better. The experiment shows that smaller or higher values do not yield better runtimes.

Interestingly, one can observe that the gradient descent outperforms the conjugate gradient method in most cases. In particular for the biharmonic operator, conjugate gradient needs slightly more time for all masks. Nevertheless, the difference between those two approaches is rather small.

Obviously, the primal-dual approach has a completely different runtime behaviour than the other approaches. For the harmonic operator, it performs bad for smaller mask densities. For a density of 1% or less, it even took more than one hour to perform a single tonal optimisation. However, its strength lies in the tonal optimisation for denser masks. With densities of 10% or higher, the primal-dual approach can outperform the other methods, at least for homogeneous diffusion inpainting. When it comes to the biharmonic operator, the primal-dual approach is a complete failure considering its runtimes. Here, the algorithm even had to be aborted after an hour for densities smaller than 4%. For all other densities, the performance is not much better.

In summary, the following results arise from this experiment. For mask densities up to 4%, the method relying on discrete Green's functions is the preferred one. For larger densities, gradient descent is better suited. Only for the harmonic operator and densities larger than 10%, the primal-dual approach should be chosen instead of the gradient descent.

4.7.2 GPU Performance

Unfortunately, the conjugate gradient approach and the approach relying on discrete Green's functions are not perfectly suited for parallel computations on the GPU. Considering the Green's approach, it typically needs more memory than available on a graphics card to store the large system matrix. Moreover, the sophisticated numerical algorithms require being computed with double precision. However, the graphics card is mainly designed for float precision, so one would have to find a good way to handle higher precisions. Lastly, both the Green's approach as well as the conjugate gradient approach rely on computing inner products. A straightforward way to evaluate these inner products would be to use a regression scheme, but this requires more elaborate implementations to obtain an efficient algorithm on such a hardware. Thus, it remains an open point to come up with an efficient GPU implementation of these two methods.

Nevertheless, we are able to compare the gradient descent algorithm and the primal-

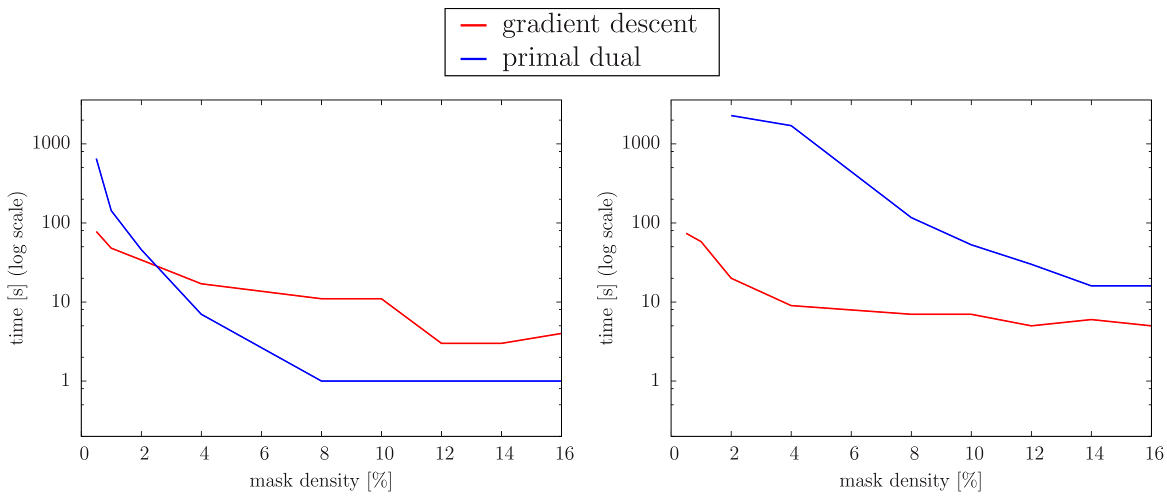

Figure 4.7. Runtime comparison of tonal optimisation strategies for various mask densities with gradient-descent and primal-dual approach on a GPU. **Left:** Harmonic inpainting operator. **Right:** Biharmonic inpainting operator.

dual approach. Both of them are perfectly suited for an implementation on the GPU. The gradient descent scheme uses an outer FED cycle length of 15. For solving the inpainting problem and for computing the Jacobian in the intermediate steps, an inner FED cycle length of 10 yields the best performance. As stopping criterion the residuum is considered for each pixel individually, allowing for an efficient evaluation on the GPU. The iterations are stopped when there is no pixel with an absolute residuum larger than 10^{-5}. On the other hand, the primal-dual algorithm stops whenever the update for any pixel has a smaller absolute value than 10^{-5}, as suggested in [93]. These tolerances allow to obtain precise results for all tested masks and the same MSE value with both approaches. The final results are depicted in Figure 4.7. Thereby, a Nvidia® GeForce GTX 460 graphics card is used. Note that the time axis is again plotted on a logarithmic scale.

Considering the primal-dual approach, the runtime is much better compared to the CPU variant. For the harmonic operator and mask densities larger than 8%, it only needs around 1 second, which is extremely fast. For less mask points, the runtime increases very fast as already seen in the CPU version. For the biharmonic inpainting operator, the performance of the primal-dual approach is much worse compared to the harmonic counterpart, and even exceeds reasonable runtimes for very small mask densities on a GPU.

The gradient descent approach on the other hand is very fast for both operators, and in particular for higher densities. It clearly outperforms the primal-dual approach for the biharmonic operator. For the harmonic operator, the gradient descent approach is better for mask densities smaller than 2%. Thus, it is the preferred method for those cases. Another observation of the gradient descent method is that for low mask densities, the acceleration of the GPU version compared to the CPU variant is not as large as for higher mask densities. The reason for this is that an FED solver is used for solving the inner linear systems of equations instead of a multigrid solver. This means that lower frequencies in the error spectrum are not removed as fast as with the multigrid solver, leading to a higher runtime in such cases where this behaviour is beneficial.

4.8 Tonal Optimisation for Nonlinear Inpainting Operators

Interestingly, we can also apply the gradient descent approach for the tonal optimisation with nonlinear inpainting operators such as EED. Here, specific challenges arise when extending it to EED-based inpainting. The nonlinearity of the EED inpainting scheme prevents closed form expressions such as (4.10) and (4.11). This is caused by the fact that the inpainting matrix $\boldsymbol{M}(\boldsymbol{u})$ is now a function depending on the inpainting $\boldsymbol{u}^k$, which itself depends on the tonal data $\boldsymbol{g}^k$ in the specified pixels. Although this concatenated mapping is formally smooth, one should keep in mind that EED has the ability to create edge-like structures. This means that in practice the problem is fairly ill-conditioned: Small local changes in $\boldsymbol{g}^k$ may have a strong global impact on $\boldsymbol{u}^k$, on $\boldsymbol{A}(\boldsymbol{u}^k)$, and on the Jacobian of the error function (4.8). Unfortunately, discrete Green's functions also do not help in this nonlinear scenario as they are currently only available for the linear case. In the future, one might think of employing them for the nonlinear case as well.

Moreover, there is no guarantee anymore that the tonal optimisation problem is strictly convex. Thus, it may have many local minimisers. It is therefore not surprising that different algorithms and even different parameter settings within the same algorithm may end up in different minimisers. Since it is difficult to design practically feasible algorithms that guarantee to find the global minimiser, we focus on a transparent and conceptually simple local optimisation strategy such as the gradient descent method. We have done a number of experiments with several variants of gradient descent approaches for tonal optimisation in EED-based inpainting. Below, the method with the best results in terms of reconstruction quality is described. Note that we have presented this approach in [1], based on first results of Mainberger using inpainting echoes [124].

We suggest to modify the gradient descent approach for (4.8) as follows. While we keep the basic structure of (4.9) with the Jacobian given in (4.10), we lack a closed form solution of type (4.11) for the Jacobian $\boldsymbol{J}(\boldsymbol{g}^k)$ that is now an unknown function of the evolving grey value data $\boldsymbol{g}^k$. As a remedy, we approximate the Jacobian by numerical differentiation:

$$(\boldsymbol{J}(\boldsymbol{g}^k))_{i,j} := \frac{\boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k + \eta\, \boldsymbol{e}_j\right)_i - \boldsymbol{r}\left(\boldsymbol{c}, \boldsymbol{g}^k\right)_i}{\eta}, \tag{4.42}$$

where i denotes the pixel index, j is the index of an individual mask point, and the parameter $\eta > 0$ represents a small grey value perturbation at the mask point j. As before, $\boldsymbol{e}_j$ is the canonical basis vector with a unit impulse in pixel j. Note that the grey values at all non-mask points do not have any influence on the inpainting result. Thus, the Jacobian does not have to be computed for those locations.

In contrast to the linear grey value optimisation strategies, we abstain from adapting the gradient descent step size α by means of exact line search or cyclic FED-based variations: We have experienced that the high sensitivity of the Jacobian w.r.t. $\boldsymbol{g}^k$ suggests to keep the gradient descent step size α fairly small to capture this dynamics in an adequate way. On the other hand, too small values for α can also result in a convergence towards a fairly poor local minimiser. Thus, in practice, we fix α to some small value.

Similar considerations apply to the parameter η in (4.42): The nonlinear dynamics of the Jacobian suggests to use small values for η in order to approximate the derivative well. On the other side, if η is too small, numerical problems can arise, since both the numerator and the denominator in (4.42) approach zero.

To show the effectiveness of tonal optimisation for EED inpainting, we perform the following experiment. We consider the original image *trui* (see Figure 2.2) and a fixed mask shown in Figure 4.8 on the left. The mask is optimised with a sophisticated method for the purpose of EED inpainting. More details on how to obtain the mask is presented in Chapter 5. First, the inpainting with the grey values of the original image at the mask points is computed. The result is shown in the middle of Figure 4.8. In the whole experiment, the EED parameters are fixed to $\lambda = 0.8$ and $\sigma = 0.7$, since this gives reconstructions of high quality. Then, a tonal optimisation is performed with different values for the parameters α and η. The resulting MSE values are presented in Table 4.3.

The best result with the parameters $\alpha = 0.01$ and $\eta = 1.0$ is shown in Figure 4.8 on the right. Although the quality of the inpainting with original grey values is already very convincing, it is possible to further improve the reconstruction by a significant amount: The MSE can be lowered from 12.62 to 10.79, which represents an error reduction of 15%. On the negative side, one has to state that the tonal optimisation in the nonlinear case can last more than a week due to the enormous effort for recomputing the Jacobian after each iteration.

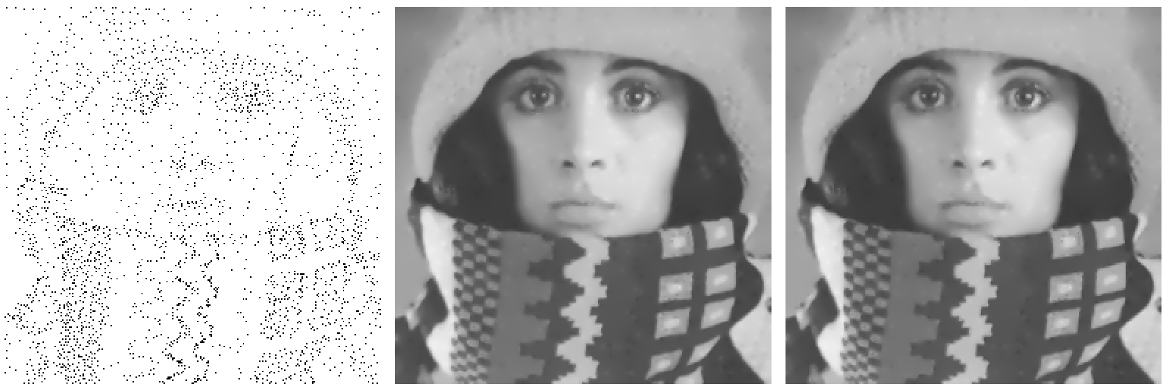

Figure 4.8. Tonal optimisation for EED inpainting. **Left:** Prescribed inpainting mask (density: 4%). **Centre:** EED inpainting with tonal values from original image *trui* (MSE: 12.62). Parameters: $\lambda = 0.8$, $\sigma = 0.7$. **Right:** Inpainting with optimised grey values (MSE: 10.79).

Table 4.3. MSE after a tonal optimisation for the EED inpainting approach with the mask from Figure 4.8. The parameter α denotes the fixed step size in each gradient descent iteration, and η is the step size used for the computation of the derivatives by means of finite differences.

	α	
η	10^{-3}	10^{-2}
1.0	10.86	**10.79**
1.5	10.93	10.80
2.0	10.88	10.80
3.0	10.88	10.81

4.9 Quantisation-aware Tonal Optimisation

After the tonal optimisation step, continuous values with floating point precision are typically obtained at the mask locations. Storing this data directly requires a lot of memory and is not very efficient. Alternatively, a better strategy is to apply a quantisation which restricts the tonal values to a finite set Q of possible values. It is clear that the inpainting quality suffers from those less optimal values at the mask points. Thus, one often has to find a good compromise between coding cost and reconstruction quality. To do so, the quantisation parameter $q := |Q|$ representing the number of admissible values allows to steer this trade-off. Since the human visual system is only able to distinguish around 40 different grey-scales, the hope is that a small q can be chosen without degrading the quality of the inpainting too much. Instead of storing the quantised tonal values directly after the quantisation, it is more efficient to encode their respective indices $1, \ldots, q$.

Uniform quantisation is the most common type of quantisation. Here, the values in Q are obtained by an equidistant sub-sampling of the range $[0, 255]$. Thereby, one can distinguish between the so-called midtread and midrise quantisations: While the midtread quantisation includes the border values 0 and 255, the midrise quantisation chooses the centres of q equally distributed intervals between 0 and 255. Thus, the midrise quantisation has a reduced range compared to the midtread quantisation. Since one is often interested in capturing the full range of an image in $[0, 255]$, the midtread quantisation will be employed in the following.

Let us now consider how the quantisation can be realised given some fixed Q. One straightforward approach is to perform the quantisation after the tonal optimisation step. This means that an optimised grey value g_i at a mask point i is projected onto the set Q by looking for the closest value. More formally, the quantised value g_i' is given by

$$g_i' = \underset{g \in Q}{\arg\min}\, |g - g_i|. \tag{4.43}$$

The drawback of this approach is that it does not yield the optimal combination of quantised grey values. As proposed in [170], one can thus additionally perform a brute-force optimisation. Thereby, the tonal data are initially projected on the set Q. Then, the individual tonal values at the mask points are iteratively shifted to the next higher or lower values in Q. The individual changes are kept whenever it improves the reconstruction quality measured in terms of the MSE. Although one obtains very good results with this approach, the naïve search strategy is not very efficient and leads to a suboptimal performance.

In order to find optimised quantised grey values in a more efficient and more elegant way, a novel improved strategy is shown in the following. It has been published in [6]. While yielding comparable results to the brute-force search, the new approach bases on the well-founded theoretical framework presented in Section 4.2, i.e. it relies on the inpainting echoes. In order to incorporate the quantisation into the optimisation algorithm, the idea is to quantise the individual grey values directly after having optimised each of them. It is clear that for the quantisation of a single tonal value, the quantised grey value given by formula (4.43) is the optimal one. The adapted algorithm is shown in Algorithm 4.6.

As we have seen in Section 4.2, the computation of the inpainting echoes requires some time. Thus, performing one quantisation-aware tonal optimisation is not very efficient. However, one can reuse the inpainting echoes for an arbitrary quantisation parameter q due to the fact that the mask remains constant. This will allow for a thorough and efficient optimisation of q, as we will see in Chapter 6.

As an example, let us consider the tonal optimisation with different values of the quantisation parameter q. We consider the image *trui* and the optimised masks which

Algorithm 4.6. Quantisation-aware Tonal Optimisation with Inpainting Echoes.

```
Input: Original image $\boldsymbol{f}$, mask $\boldsymbol{c}$, mask indices $K$, set of quantisation values $Q$.
Initialisation: Inpainting $\boldsymbol{u} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})$, grey values $\boldsymbol{g}^0 = \boldsymbol{C}\boldsymbol{f}$, $\varepsilon > 0$.
Compute:
  For all $i \in K$:
      Compute the inpainting echo $\boldsymbol{b}_i = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{e}_i)$.
  Repeat for $k \geq 0$:
      For all $i \in K$:
        1. Compute the correction term $\alpha^k = \frac{\boldsymbol{b}_i^\top(\boldsymbol{f}-\boldsymbol{u}^k)}{|\boldsymbol{b}_i|^2}$.
        2. Update the grey value $g_i^{k+1} = g_i^k + \omega^k \cdot \alpha$.
        3. Apply quantisation: $g_i' = \operatorname*{arg\,min}_{g \in Q} |g - g_i^{k+1}|$.
        4. Update inpainting $\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \alpha' \cdot \boldsymbol{b}_i$ with $\alpha' = g_i' - g_i^k$.
  until the stopping criterion $\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^k\|_2^2 < \varepsilon$ is fulfilled.
Output: Inpainting $\boldsymbol{u}$ containing optimised quantised grey values at the mask locations.
```

have been depicted in Figure 4.1 for the harmonic inpainting operator and in Figure 4.2 for the biharmonic one. Without any quantisation, the MSE of the tonal optimised inpainting for those masks is 27.24 and 16.73, respectively. In Figure 4.9, the result of tonal optimisation with quantisation is shown. The blue line represents the MSE of the inpainting which arises from a subsequent quantisation of the continuous tonal optimised values. The red line shows the MSE obtained by the quantisation-aware tonal optimisation. For a reference, the black line indicates the lower bound inpainting error given by a tonal optimised inpainting without quantisation.

In the plots, one can nicely see the influence of the quantisation on the inpainting quality. For larger values of q, the MSE is very close to the one of the continuous result. On the other hand, for smaller choices of the quantisation parameter, the error exponentially increases for both quantisation strategies. It is also obvious that it pays off to incorporate the quantisation into the optimisation process. Especially for smaller values of the quantisation parameter, the final MSE is much smaller with this more sophisticated approach.

To get a visual impression on the influence of a quantisation, Figure 4.10 shows the results with a quantisation parameter $q = 16$. Also here, one can see the superior result given by the quantisation-aware tonal optimisation compared to the naïve approach with a subsequent quantisation of the continuous values. It is hard to see

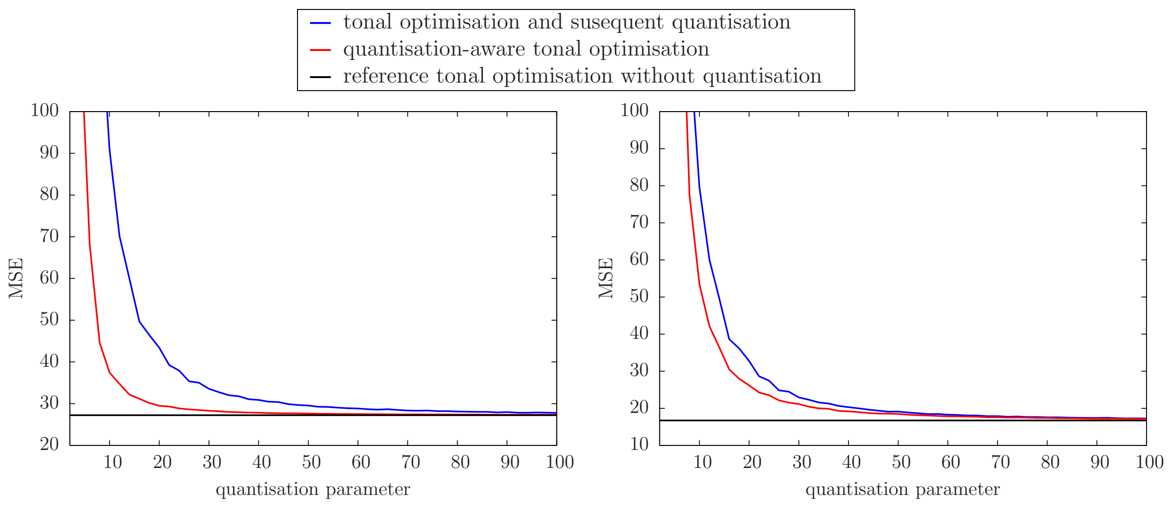

Figure 4.9. Different tonal optimisation strategies with varying quantisation parameter q. **Left:** Harmonic inpainting operator. **Right:** Biharmonic inpainting operator.

the difference between the inpainting without quantisation and the inpainting with quantised values obtained with the quantisation-aware approach. This quality is remarkable when considering that only 16 different grey values are used at the mask points for computing the inpainting. In contrast, the inpainting with subsequently quantised values reveals obvious degradations.

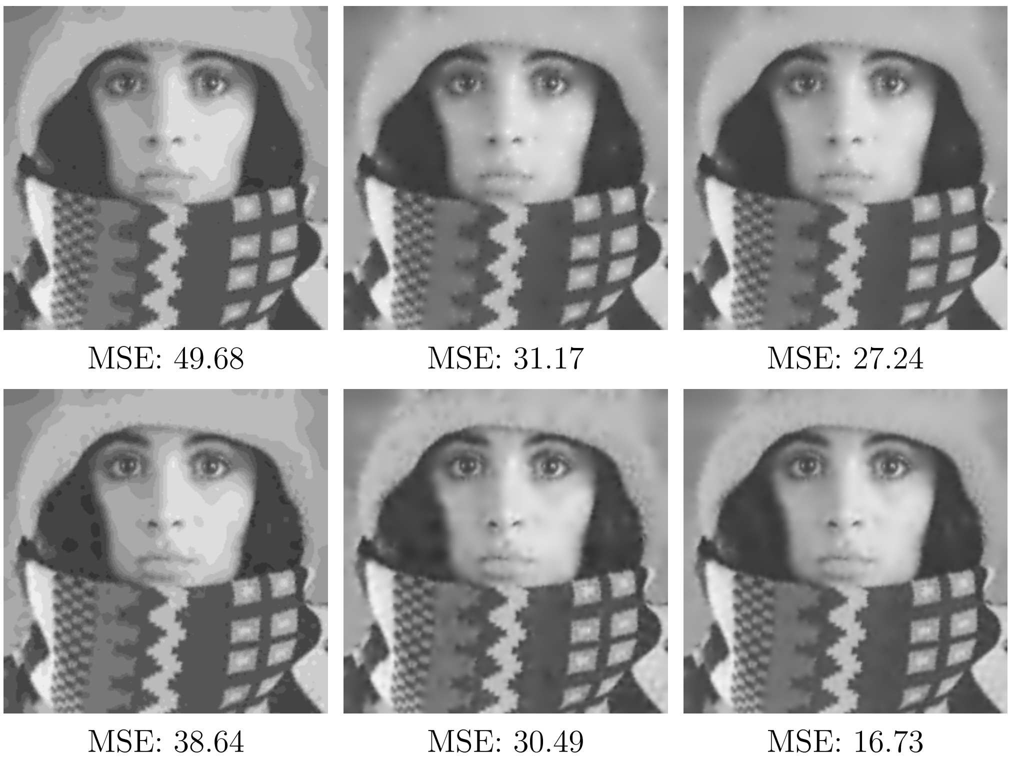

Figure 4.10. Tonal optimisation with quantisation ($q = 16$). **Top row:** Harmonic inpainting for optimised mask shown in Figure 4.1. **Bottom row:** Biharmonic inpainting for optimised mask shown in Figure 4.2. **Left:** Inpainting after quantising continuous tonal optimised values. **Centre:** Quantisation-aware tonal optimisation. **Right:** Tonal optimised inpainting without quantisation.

4.10 Conclusion

Tonal optimisation is a central part of PDE-based image compression, and it is important to have efficient methods to perform this task. Considering the presented strategies, we have seen that there is no best method, but they all have advantages in certain scenarios.

The approach relying on inpainting echoes has the drawback that the computation of the echoes is a relatively large overhead. Moreover, the echoes also lead to rather high memory requirements. Nevertheless, this approach also has advantages. Firstly, it allows to incorporate a quantisation in the optimisation process leading to better results compared to a naïve subsequent quantisation of the continuous optimal values. Secondly, the echoes can be reused whenever a tonal optimisation needs to be performed many times for the same mask. If no high accuracy is needed, the echoes can be approximated due to their limited local influence.

If continuous optimal values are desired, recall that there is a unique solution of the convex tonal optimisation problem with linear inpainting operators. In this case, the runtime is the most important criterion to judge the performance of a method. To select the best approach, we have seen that this choice highly depends on several factors such as the mask density, the inpainting operator and the hardware.

The gradient descent approach has a very stable performance, and does not depend much on the mask density. Compared to the other methods, it performs best with the harmonic operator for medium densities, and is the favoured choice with the biharmonic operator for densities larger than 4%. Moreover, it is the only method which enables a tonal optimisation for nonlinear inpainting operators as well, and is well suited for a GPU implementation. The conjugate gradient approach has a similar performance as the gradient descent for the linear operators, but is slightly slower and only has a limited potential for a parallel processing.

The primal-dual approach is very fast for harmonic inpainting and higher mask densities. However, it has a slow convergence for smaller mask densities or other operators, which makes it impracticable in some cases. It also allows for a GPU implementation, leading to very fast results for homogeneous diffusion inpainting and larger densities.

Lastly, the approach with the discrete Green's functions bases on the theory from Chapter 3. Due to the close connection to the sparsity, it is is the favoured choice for small mask densities. A pleasant property is that the runtime does not depend on the inpainting operator, and can easily be extended to even higher orders by simply using the appropriate power of the eigenvalues. Unfortunately, especially for higher order operators, numerical inaccuracies may occur for large mask densities.

Chapter 5

Spatial Optimisation

5.1 Motivation

In Chapter 4, we have seen that a clever selection of the tonal values tremendously increases the reconstruction quality. We have observed in Section 2.4 that the inpainting also highly depends on the selected mask locations. Consequently, the following question naturally arises: Given a certain prescribed density of points, how does an optimal mask configuration look like? As for the tonal values, the quality of a mask is judged by the quality of the corresponding inpainting, measured in terms of the MSE.

Due to the discrete nature of the problem, it seems tempting to try all different combinations for finding the best mask, and choose the one yielding the best reconstruction. Unfortunately, this is not feasible in practice due to the enormous amount of possible mask configurations, as already mentioned in Section 1.2.2. In fact, choosing a good mask is an even harder optimisation problem compared to the tonal optimisation problem. The reason for this is that there is a nonlinear dependency of the inpainting on the mask.

The joint influence of the mask and the tonal values make it indispensable to treat those two ingredients together. In particular, we judge the quality of a mask by evaluating its overall potential to reconstruct an image, including tonal optimised values at the selected mask locations. If we only relied on the values of the original images at the mask points when computing the inpainting, the real reconstruction potential of a mask would not be considered.

In this chapter, we will discuss different approaches to find an optimised mask. We mainly focus on the linear inpainting operators, but also focus on nonlinear operators later. To compare the individual methods, we consider the following sample problem for all of them: For the image *trui* from Figure 2.2, we seek a mask containing 4% of all pixels.

This chapter is structured as follows. We start by reviewing an existing approach which bases on an optimal control framework in Section 5.2. Afterwards in Section 5.3, probabilistic approaches are considered. We discuss an analytic approach relying on the continuous theory of Belhachmi *et al.* in Section 5.5. Afterwards, strategies using the theory of discrete Green's functions are presented in Section 5.6. Section 5.7 considers the problem of finding optimal masks for nonlinear inpainting operators. Finally, a conclusion is given in Section 5.8.

5.2 Optimal Control

For homogeneous diffusion inpainting, an optimal control framework has been proposed for the purpose of finding a mask in [92]. This is achieved by searching for a suitable minimiser of the following energy:

$$\begin{aligned} &\operatorname*{arg\,min}_{\boldsymbol{c},\boldsymbol{u}\in\mathbb{R}^{|\Gamma|}} \tfrac{1}{2}\|\boldsymbol{u}-\boldsymbol{f}\|_2^2 + \lambda\|\boldsymbol{c}\|_1 + \tfrac{\varepsilon}{2}\|\boldsymbol{c}\|_2^2\,, \\ &\text{subject to: } \boldsymbol{C}(\boldsymbol{u}-\boldsymbol{f}) - (\boldsymbol{I}-\boldsymbol{C})\boldsymbol{A}_N\boldsymbol{u} = \boldsymbol{0}. \end{aligned} \tag{5.1}$$

By minimising this energy, a compromise is found between inpainting quality and mask sparsity. While the quality of the inpainting $\boldsymbol{u}$ is measured by the Euclidean distance to the original image $\boldsymbol{f}$, the sparsity of the mask $\boldsymbol{c}$ is approximated by its ℓ_1-norm. With the help of the parameter λ, the density of the mask can be steered. The linkage between the mask and the inpainting is achieved by enforcing the inpainting equation to hold true in a side constraint. The additional term $\frac{\varepsilon}{2}\|\boldsymbol{c}\|_2^2$ only exists for technical reasons, and can as well be left out without having an influence on the solution [51].

Instead of a binary mask, the constraint is relaxed to allow a real-valued mask. We have already discussed this in Section 2.6, and as it turned out, one has to ensure that the mask values stay within the range $[0, \frac{8}{7}]$ to guarantee that there exists a solution of the inpainting problem. To obtain the desired binary mask in the end, the continuous mask has to be thresholded.

The main issue for finding a minimiser of (5.1) is that the constraint makes the energy non-convex. As proposed in [92], the minimiser can be found by replacing the energy with a series of convex problems. This is achieved by linearising the constraint with respect to the unknowns $\boldsymbol{u}$ and $\boldsymbol{c}$. Each of the convex problems is then solved with the primal-dual approach. For more details, please see [92].

In this optimal control approach, it seems that the mask is optimised without paying attention to a potential tonal optimisation. In particular, after thresholding the continuous mask $\boldsymbol{c}$ resulting from the optimisation problem (5.1), a subsequent optimisation of the tonal values is usually done in practice. Thus, it seems that the spatial and tonal optimisation is treated independently. It could be shown in [93] that the optimisation of the continuous mask values $\boldsymbol{c}$ is equivalent to a tonal optimisation.

Thus, the grey values are inherently optimised within the optimal control framework, too. This explains the very good quality of the resulting masks.

Besides inpainting with the harmonic operator, the optimal control approach can also be extended to biharmonic inpainting in a straightforward way. Therefore, one mainly has to replace the inpainting operator $\boldsymbol{A}_N$ by its biharmonic counterpart $\boldsymbol{B}_N$. Additionally, it is possible to incorporate the diagonal preconditioning into the primal-dual solver, similar to what we have done for the tonal optimisation in Section 4.5. This leads to a remarkable acceleration. Both extensions have been presented in Chen *et al.* [51]. In this work, the method is also applied for finding a mask for inpainting with TV regularisation, and the iPiano method proposed in [148] is considered as numerical solver.

There is one major problem of this approach: Due to the intransparent dependency of the parameter λ to the final mask density, it is not possible to specify a desired number of mask points from the beginning. Since the final mask density for the same λ changes for two different images, it is also not possible to set up a look-up-table once. One possibility to obtain a desired density is to perform a so-called grid search. Thereby, it is assumed that the relation between λ and the mask density is a convex function. The grid search algorithm then works as follows. First, for a fixed number of uniformly distributed values of λ within some larger range, the minimisers of the optimisation problem (5.1) are computed. One has to make sure that the unknown λ leading to the desired density lies within this initial range, so it makes sense to choose the range generously. Then, one chooses the two values of λ whose corresponding mask densities are closest to the desired one. This defines a new smaller search range, and one continues by performing the optimisation with updated samples of λ. This procedure is repeated iteratively until the desired mask density is obtained.

It is clear that the grid search procedure is rather inefficient due to the large number of optimisations that need to be done until a desired mask density is found. Furthermore, experiments show that the method converges much slower the larger λ is chosen. Altogether, it can take days until a mask with a specific density is found. Thus, for speeding things up, we implemented the optimal control approach on a graphics hardware. This helps to reduce the computation time from days to minutes or hours, depending on the mask density and how fast the unknown parameter λ can be found to match the desired mask density.

Let us consider some example results for the image *trui*. For both harmonic and biharmonic inpainting, the masks with a prescribed density of 4% are depicted in Figure 5.1. On the GPU, they can be obtained within some minutes when knowing the actual parameter λ. Obviously, the mask points tend towards the image edges. This can be explained with the continuous theory of Belhachmi *et al.* [23] which we will consider in more detail in Section 5.5.

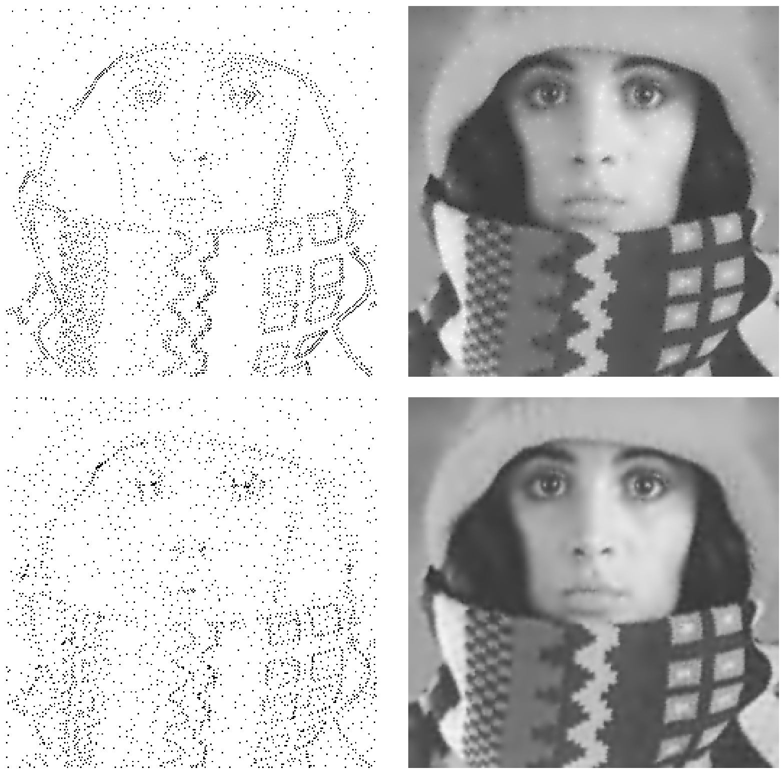

Figure 5.1. Spatial optimisation with optimal control approach. **Top row:** Optimised mask (density: 4%) and corresponding tonal optimised inpainting for harmonic inpainting ($\lambda = 0.006$). MSE: 22.41. **Bottom row:** Same for biharmonic inpainting ($\lambda = 0.0042$). MSE: 16.14.

5.3 Probabilistic Approaches

Probabilistic sparsification and densification are strategies which rely on a greedy strategy to find a good mask. In contrast to the optimal control approach, they have the advantage that they can be used for any inpainting operator. In particular, one is not restricted to linear inpainting operators. Moreover, the probabilistic approaches benefit from the possibility that the desired mask density can be specified explicitly. Thus, there is no need for tuning any parameter to obtain a certain amount of mask points. The sparsification has first been proposed in [4], and a more thorough analysis is carried out in [1]. The densification, on the other hand, has first been introduced in [2], Section 3.2. Is has been further extended and investigated in the context of tree-like mask structures in [5, 6]. In the following, the probabilistic sparsification is presented first, and the densification afterwards.

Algorithm 5.1. Probabilistic Sparsification.

```
Input: Original image $\boldsymbol{f}$, fraction $p$ of mask pixels used as candidates, fraction $q$ of
  candidate pixels that are removed in each iteration, desired mask density $d$.
Initialisation: Set $\boldsymbol{c} = \boldsymbol{1}$, $K = \Gamma$.
Compute:
  Do:
    1. Choose a random candidate set $T$ containing $p \cdot |K|$ pixel indices from $K$.
    2. For all $i \in T$ set $c_i = 0$.
    3. Compute the inpainting $\boldsymbol{u} = \boldsymbol{r}\,(\boldsymbol{c}, \boldsymbol{f})$.
    4. For all $i \in T$ compute the local error $e_i = (u_i - f_i)^2$.
    5. Remove all $i$ with the $(1-q) \cdot |T|$ largest errors from $T$ and reassign $c_i = 1$.
    6. Remove the remaining indices $i \in T$ from $K$ and clear $T$.
  until $|K| = d \cdot |\Gamma|$.
Output: Optimised mask locations $K$.
```

5.3.1 Probabilistic Sparsification

Starting with a full mask, where every pixel is chosen as a mask point, probabilistic sparsification iteratively removes the least significant mask pixels until a desired density d is reached. To do so, in each step, a fraction p of candidate pixels from the current mask is selected. These pixels are removed from the mask, and an inpainting with this updated mask is calculated. The significance of a candidate pixel is then estimated by computing the local error, i.e. the squared grey value difference of the inpainted and original image in this pixel. Then, the fraction q of the candidates that exhibit the smallest local error are permanently removed from the mask, and the remaining fraction $(1-q)$ of the candidates are inserted back again. A detailed description of the method is given in Algorithm 5.1.

Note that depending on p, multiple points are removed at once. As a mask point theoretically has a global influence on the inpainting result, the alleged measured significance of a mask point also includes the influence of the other removed points. One possible mitigation of this issue is to guarantee for some distance of the selected mask points in each iteration, but this introduces a bias which could lead to suboptimal local minimum. Moreover, it seems tempting to choose p very small to obtain only the influence of some few mask points. However, there is a high chance that only important points are selected, from which some are removed. In [1], the results with different choices of the parameter have been considered in an experiment with several original images. Thereby, a value of p between 0.1 and 0.3 gives the best results.

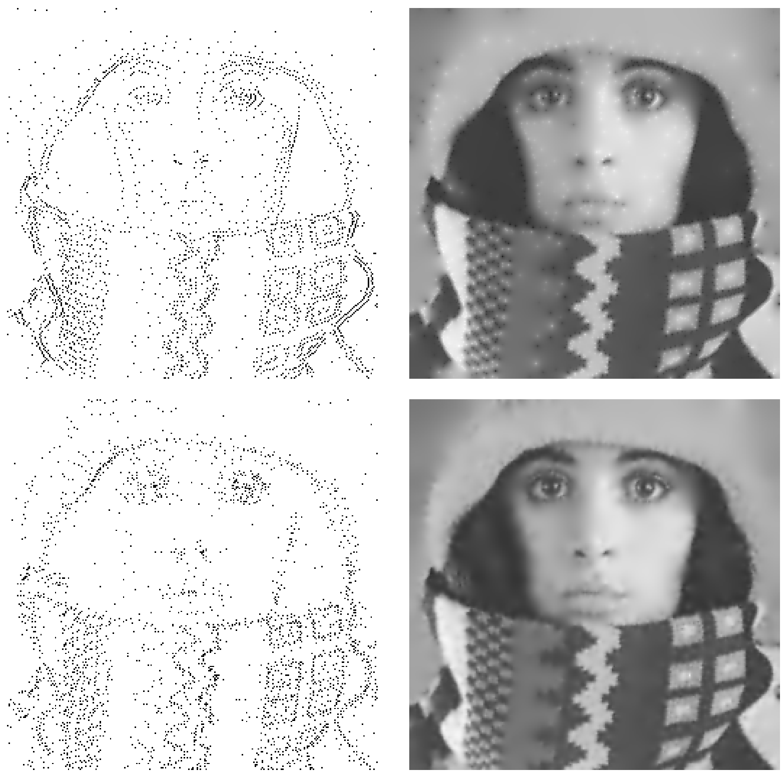

Figure 5.2. Spatial optimisation with probabilistic sparsification. **Top row:** Optimised mask (density: 4%) and corresponding tonal optimised inpainting for harmonic inpainting ($p = 0.3$, $q = 10^{-6}$). MSE: 36.04. **Bottom row:** Same for biharmonic inpainting ($p = 0.005$, $q = 10^{-6}$). MSE: 22.39.

Unfortunately, this choice highly depends on the image which makes it hard to universally fix this parameter to a specific value. When it comes to the parameter q, it is clear that the best would be to choose a small value. This is also confirmed by the experiments in [1]. The only drawback is that the whole process then becomes very time-consuming.

Example results for the image *trui* are shown in Figure 5.2. Although the masks look very similar to the ones obtained with the optimal control approach, the reconstruction quality is worse with the probabilistic sparsification. After having a closer look, one can see that the latter masks contain less points in homogeneous regions, as for example at the upper right part of the image. In contrast, the masks of optimal control have more points at those locations. This indicates that it might be good to have points in such regions, too.

Algorithm 5.2. Probabilistic Densification.

```
Input: Original image $\boldsymbol{f}$, fraction $p$ of mask pixels used as candidates, fraction $q$ of
  candidate pixels that are added in each iteration, desired mask density $d$.
Initialisation: Set $\boldsymbol{c} = \boldsymbol{0}$, $K = \emptyset$ and $\boldsymbol{u}$ to the mean value of $\boldsymbol{f}$.
Compute:
  Do:
    1. Choose a random candidate set $T$ containing $p \cdot |K|$ pixel indices from $K$.
    2. For all $i \in T$ compute the local error $e_i = (u_i - f_i)^2$.
    3. Insert all $i \in T$ with the $q \cdot |T|$ largest errors into $K$ and assign $c_i = 1$.
    4. Update the inpainting $\boldsymbol{u} = \boldsymbol{r}\,(\boldsymbol{c}, \boldsymbol{f})$, and clear $T$.
  until $|K| = d \cdot |\Gamma|$.
Output: Optimised mask locations $K$.
```

5.3.2 Probabilistic Densification

Probabilistic sparsification is designed to start with a full mask and gradually removes points until the desired density is obtained. One can imagine that this requires more time the sparser the final mask shall be. In particular in a compression framework, typical mask densities range between 0 and 20 percent, which is a long way do go for the sparsification. Alternatively, one can think of starting from the other end. This is done in the probabilistic densification. Here, an empty mask is used in the beginning, and points are iteratively inserted at locations where it is most urgent. More precisely, in each step, a fraction p of candidate pixels from the current non-mask points is selected. Then, the local error is computed at all those locations by means of the squared difference in the grey values. Then, a fraction q all the candidate pixels with the largest error are added to the mask, and an inpainting with this updated mask is calculated. This procedure is repeated until the desired mask density is reached. A complete overview of the method is given in Algorithm 5.2.

In contrast to the sparsification approach, the influence of a mask pixel on the inpainting is only considered indirectly in the densification: After a mask point is inserted and the inpainting is updated, the significance of a point becomes visible when computing the error for neighbouring points in the next iterations. In the sparsification, the significance is estimated before removing a mask point. However, as already discussed, this is also only an approximation. In contrast to sparsification, the danger of not including an important mask point is smaller as it will be inserted with a very high probability in the beginning. Furthermore, a big advantage of the densification is that smaller densities which are often desired in compression contexts can be obtained much faster.

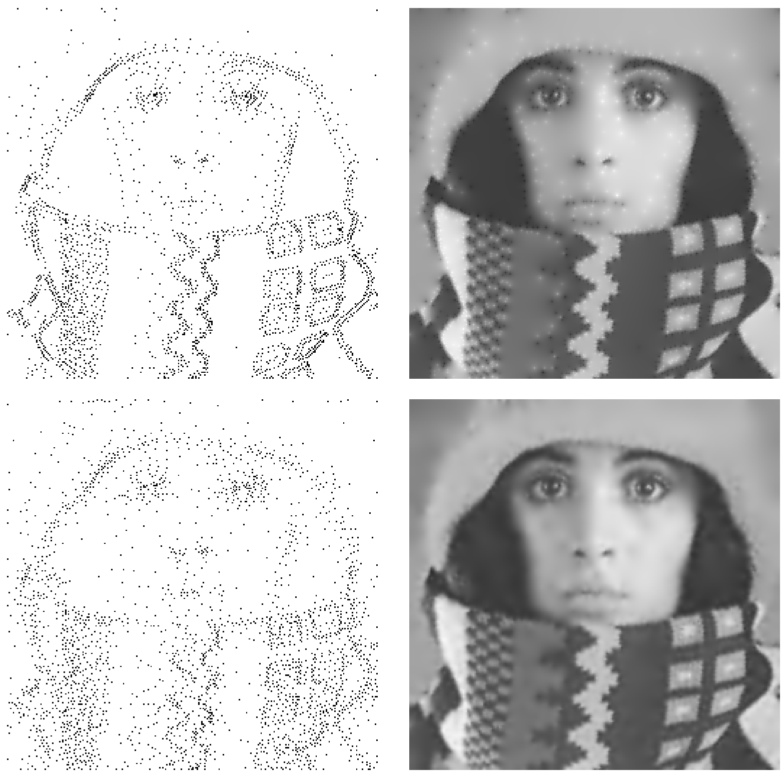

Figure 5.3. Spatial optimisation with probabilistic densification. **Top row:** Optimised mask (density: 4%) and corresponding tonal optimised inpainting for harmonic inpainting ($p = 0.2$, $q = 10^{-6}$). MSE: 29.92. **Bottom row:** Same for biharmonic inpainting ($p = 0.1$, $q = 10^{-6}$). MSE: 19.76.

Again, example results are in Figure 5.3. Although one can hardly see a difference of the masks between the sparsification and densification, the reconstructions are better for the densification. This however cannot be generalised as sparsification can also give better results than densification for certain images. Nevertheless, one can conclude that the probabilistic approaches give reasonably good results, while not being optimal.

5.3.3 Limitations

There are several reasons why the probabilistic approaches cannot guarantee to yield the optimal mask. As already indicated, one issue is that both methods only approximate the importance of a mask point on the inpainting. While probabilistic sparsification checks how leaving out certain mask points influence the inpainting, probabilistic densification seeks for missing points by considering the reconstruction error. It is clear that this cannot yield the optimal result. Even incorporating a smoothing of the image during the evolution does not improve the mask quality. Another reason is the

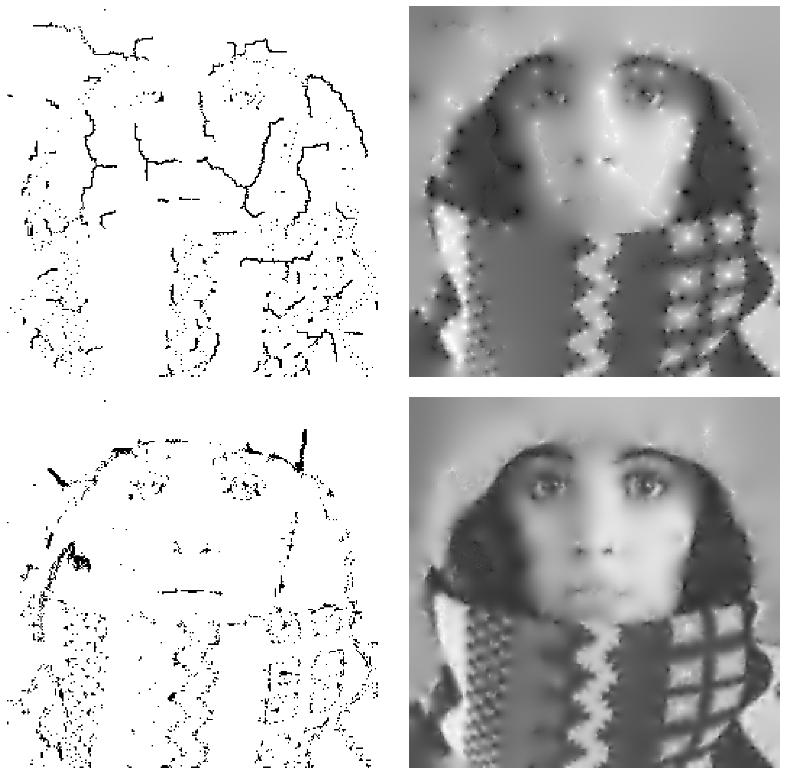

Figure 5.4. Spatial optimisation with probabilistic densification incorporating tonal optimisation after every mask point insertion. **Top row:** Optimised mask (density: 4%) and corresponding tonal optimised inpainting for harmonic inpainting ($p = 0.1$, $q = 10^{-6}$). MSE: 165.63. **Bottom row:** Same for biharmonic inpainting ($p = 0.1$, $q = 10^{-6}$). MSE: 69.63.

global interdependence between all mask pixels which is not completely accounted for in these greedy approaches. Once a point is removed or inserted, there is no possibility to revert this decision. Thus, especially at stages where only few mask pixels are present, important points might be missed, so that one ends up in a suboptimal local minimum. In this regard, the probabilistic component also plays a major role. If exclusively important points are selected for the candidate set, only imperfect results can be obtained.

Another potential issue of both approaches is that no tonal optimisation is respected during the mask evolution. The methods only rely on the original tonal values at the mask points for obtaining the intermediate inpainting results. However, due to a tonal optimisation, a mask point could potentially be used at a different location when the grey value can compensate for its absence. One straightforward idea would thus be to perform a tonal optimisation step after each inpainting. This way, the grey value optimisation is directly incorporated into the mask optimisation process. Unfortunately, this strategy is not yielding good results at all. For example, Figure 5.4 shows the result of probabilistic densification where the tonal optimisation is performed after each step. The mask points are not distributed nicely over the image, but they are gathered along some of the edges. The corresponding inpainting quality is hence not good.

Even though this result surprises in the beginning, the bad performance can be explained easily. After inserting mask points at certain locations, the tonal optimisation modifies the corresponding grey value such that the overall inpainting quality is best. However, this global optimality is at the cost of having a bad local approximation. Thus, instead of improving the local inpainting quality at those locations where the mask points were intended for, the error is steadily large there. Consequently, new mask points will be inserted close the the previous locations in subsequent iterations. After repeating this procedure, one finally obtains a clustering of the mask points along edges where the error remains large at all times.

This experiment shows that it does not necessarily pay off to optimise the tonal values while selecting optimal mask locations. In particular in earlier stages, an optimisation of the grey values hinders a good spatial optimisation. Instead, using the original grey values at the mask points during the evolution seems to have a smoothing effect which positively affects the spatial optimisation process. This observation has also been confirmed in related experiments in that context.

5.4 Nonlocal Pixel Exchange

As discussed, both probabilistic sparsification and densification are not guaranteed to find an optimal solution. Thus, a method called nonlocal pixel exchange that allows to further improve the results of any previously obtained mask is presented in the following. While this method has been proposed in [4], it has been further analysed in [1]. The nonlocal pixel exchange starts with a sparse and possibly suboptimal mask which already has the desired density. In each step, it randomly selects a set of m non-mask points as candidates. The candidate that exhibits the largest local error is then exchanged with a randomly chosen mask pixel. If the inpainting result with the new mask is worse than before, the exchange is reverted. Otherwise the new mask is kept. Algorithm 5.3 shows the details of the nonlocal pixel exchange method.

By construction, the nonlocal pixel exchange can only improve the result. This algorithm always converges towards an optimal solution in terms of an exchange of two pixels. Since we exchange at each iteration the same number of candidate pixels it follows that this approach is not equivalent to an exhaustive search through all possible combinations. Thus, one cannot guarantee convergence towards the global minimum. Nevertheless, the algorithm is able to find very good results on the expense of taking a very long time. As a result of the extensive experiments which have been conducted in [1], choosing the parameter m between 20 and 30 gives the best results

Algorithm 5.3. Nonlocal Pixel Exchange.

```
Input: Original image $\boldsymbol{f}$, mask $\boldsymbol{c}$, mask indices $K$, size $m$ of candidate set.
Initialisation: Set $\boldsymbol{u} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})$, $\boldsymbol{c}^{\text{new}} = \boldsymbol{c}$.
Compute:
  Do:
    1. Randomly select $m$ pixel indices $i$ from $\Gamma \setminus K$.
    2. For all $i \in T$ compute the local error $e_i = (u_i - f_i)^2$.
    3. Randomly choose one $j \in K$ and set $c_j^{\text{new}} = 0$.
    4. For the index $i$ with the largest $e_i$, set $c_i^{\text{new}} = 1$.
    5. Compute $\boldsymbol{u}^{\text{new}} = \boldsymbol{r}(\boldsymbol{c}^{\text{new}}, \boldsymbol{f})$.
    6. If $\|\boldsymbol{u}^{\text{new}} - \boldsymbol{f}\|_2 < \|\boldsymbol{u} - \boldsymbol{f}\|_2$
           Set $\boldsymbol{u} = \boldsymbol{u}^{\text{new}}$, $\boldsymbol{c} = \boldsymbol{c}^{\text{new}}$ and update $K$ accordingly.
       else
           Reset $\boldsymbol{c}^{\text{new}} = \boldsymbol{c}$.
  until the maximum number of iterations is reached.
Output: Optimised mask locations $K$.
```

for the tested images. Of course, the optimal value can vary for different images.

Moreover, one can see in Figure 7 in [1] that the mask improves a lot during the first 5000 iterations. Afterwards, the error decreases less and less the more iterations are performed. Nevertheless, one can still see improvements even after 500000 iterations. Furthermore, it is shown that it pays off to have a better mask as initialisation, as for example the one obtained with the probabilistic approaches. This increases the convergence towards the optimum. Again, a tonal optimisation is not considered in the optimisation process. This could also be achieved by optimising the grey values when computing the inpainting in every iteration. Unfortunately, this is typically not feasible due to the large runtime.

Starting with the mask obtained by the probabilistic sparsification, the nonlocal pixel exchange yields the results in Figure 5.5. Obviously, the error can be reduced enormously, and the reconstructions become closer to the ones of the optimal control approach. Nevertheless, one can see that all the greedy approaches cannot compete with the optimal control approach. Hence, theoretically more profound approaches will be considered in the next Sections.

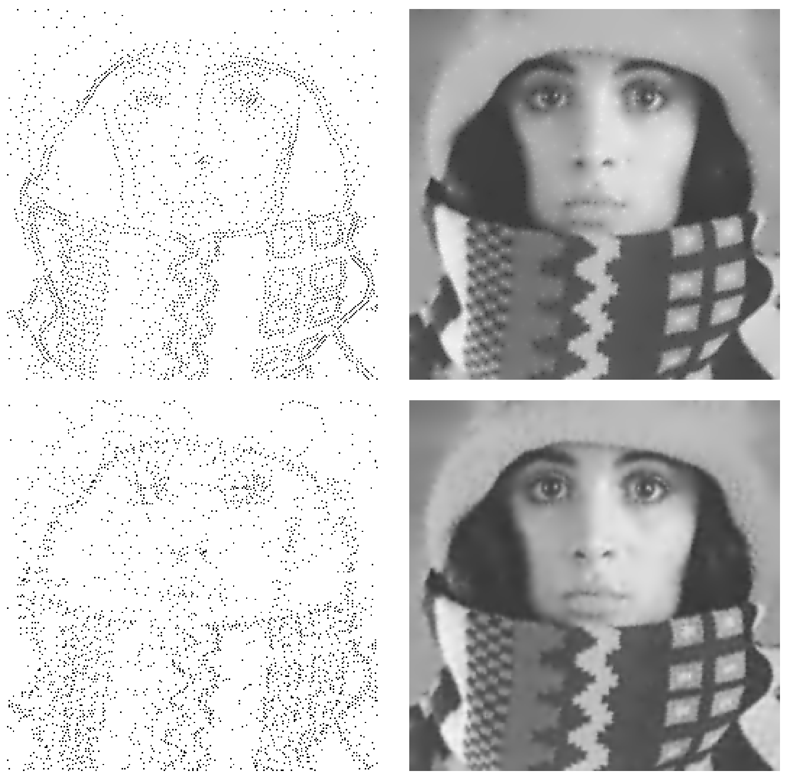

Figure 5.5. Spatial optimisation with nonlocal pixel exchange. Initialisation is mask from probabilistic sparsification. **Top row:** Optimised mask (density: 4%) and corresponding tonal optimised inpainting for harmonic inpainting ($m = 20$). MSE: 27.24. **Bottom row:** Same for biharmonic inpainting ($m = 10$). MSE: 16.73.

5.5 Analytic Approach

Belhachmi *et al.* relied on the mathematical theory of shape optimisation for finding the optimal mask points for inpainting [23]. Thereby, the topological properties of given objects are optimised. For homogeneous diffusion inpainting, the authors could show that the density of the data points should be chosen as an increasing function of the Laplacian magnitude of the original image $\boldsymbol{f}$. However, this optimality result returns a continuous density function instead of a discrete pixel mask. It is not obvious how one can obtain a binary approximation of this continuous density function. This problem is also discussed in [23]. Thus, different variants have been proposed in recent years, and are commonly called the analytic approach. After presenting them in the following, we will see that there is much more potential behind this theory.

5.5.1 Previous Approaches

Belhachmi *et al.* themselves have suggested the following strategy to obtain a point mask based on the Laplacian magnitude. First, a Gaussian presmoothing with a standard deviation of σ is applied to $\boldsymbol{f}$ to obtain $\boldsymbol{f}_\sigma$. This is a common procedure in image processing to ensure the differentiability of the data. Then one computes the Laplacian magnitude $|\boldsymbol{A}_N \boldsymbol{f}_\sigma|$ for all pixels, and obtains a density function $\boldsymbol{\mu} = c_0|\boldsymbol{A}_N \boldsymbol{f}_\sigma|$ depending on the Laplacian magnitude and some scaling parameter $c_0 \in \mathbb{R}$. Note that the scalar operations such as the absolute value and the multiplication are meant component-wise. The value of c_0 has to be determined such that $\boldsymbol{\mu}$ represents a probability density function, i.e. the mean of $\boldsymbol{\mu}$ shall represent the desired mask density d. Finally, the classical error diffusion method of Floyd and Steinberg [73] is used to obtain a binary point mask from the density distribution.

In [4], it is proposed to employ the more sophisticated electrostatic halftoning [168] instead of the Floyd and Steinberg dithering. In principle, any dithering algorithm that preserves the average grey value can be applied to transfer the density function to a binary mask. However, it could be demonstrated in [1] that preferring this approach over simpler dithering approaches leads to superior results in terms of the inpainting quality. Nevertheless, since any dithering method introduces errors, it remains an open question if this is the most suitable approach to discretise the continuous optimality result.

Moreover, the theory of Belhachmi *et al.* demands the data points to be chosen as an increasing function of the Laplacian magnitude. However, the optimal increasing function is not known, and depends on the details of the underlying model. One possibility is to assume this function to be the identity function of the Laplacian magnitude, as it is done in [23]. An alternative strategy is pursued in [1], where a power function is assumed. To this end, an additional parameter $p > 0$ is introduced which allows to tune the density of the selected points in homogeneous regions. This variant can also be motivated from the original paper [23], and leads to the modified density $\boldsymbol{\mu} = c_0|\boldsymbol{A}_N \boldsymbol{f}_\sigma|^p$.

An overview of the analytic approach including the electrostatic halftoning and the modified density function is summarised in Algorithm 5.4. Unfortunately, there exists no theory for higher order inpainting operators such as the biharmonic one so far.

In order to find good values for the parameters p and σ, a joint grid search can be performed to obtain the best possible reconstruction. To this end, the grid search explained for only one parameter is extended to multiple parameters in a straightforward way. In [1], this reconstruction is obtained by computing the inpainting with the values of the original image $\boldsymbol{f}$ as Dirichlet condition. As a result, the analytic approach is able to yield good masks. Although the quality of the inpainting with these masks is slightly worse than the ones obtained with probabilistic sparsification, the advantage of this approach is that it relies on a proper theoretical framework. Additionally, it is much faster than all other mask optimisation approaches that we

Algorithm 5.4. Analytic Approach.

```
Input: Original image $\boldsymbol{f}$, Gaussian standard deviation $\sigma$, exponent $p$, desired pixel density $d$.
Compute:
    1. Perform Gaussian presmoothing with standard deviation $\sigma$: $\boldsymbol{f}_\sigma = K_\sigma * \boldsymbol{f}$.
    2. Compute the the Laplacian magnitude $\boldsymbol{a} := |\boldsymbol{A}_N \boldsymbol{f}_\sigma|$.
    3. Calculate the constant $c_0$ as
           $c_0 = \frac{d}{\text{mean}(\boldsymbol{a}^p)}$,
       to obtain the probability density function $\boldsymbol{\mu} = c_0 \cdot \boldsymbol{a}^p$.
    4. Apply electrostatic halftoning on $\boldsymbol{\mu}$ to obtain $\boldsymbol{c}$.
Output: Optimised mask $\boldsymbol{c}$.
```

have discussed so far. In particular, the approach is even real-time capable if a fast dithering method is used. However, rather than on speed, the focus of the present work is to maximise the quality of the reconstruction. For this reason, we stick to the electrostatic halftoning for the dithering. For speeding it up, a GPU implementation is employed[1].

In order to evaluate the performance of the discussed analytic approach, a mask is found for the test image *trui*. Again, the target density is 4%. The parameters σ and p which yield the smallest error are $\sigma = 1.6$ and $p = 0.8$. The resulting mask and the corresponding tonal optimised inpainting is shown in Figure 5.6. Compared to the results of the previous approaches the obtained MSE of 42.0 is not very convincing yet. Luckily, there is much more potential behind the theory of Belhachmi *et al.* as we will see in the following.

5.5.2 Improved Variants

Considering the analytic approach in more detail, one can observe that there are still some aspects which can be improved. The hope is to obtain much better masks by a better design and a tuning of all ingredients. Different possibilities in this regard are presented in the following.

Incorporating Tonal Optimisation

In general, we are interested in finding a mask which is able to yield the best possible inpainting. As mentioned before, this would include the potential given by the tonal optimisation. Thus, it seems natural to take this into account when optimising the

[1]GPU implementation of electrostatic halftoning courtesy of Dr. Christian Schmaltz.

parameters p and λ: Instead of evaluating the inpainting with the original grey values in the grid search, it makes sense to consider the result of the tonal optimisation. In other words, the parameters are chosen in a way such that they yield the best reconstruction *after* a tonal optimisation. Although this increases the runtime, the hope is to obtain a mask with higher overall reconstruction potential. Considering the example, the MSE can be reduced from 41.51 to 35.68 after a tonal optimisation. The optimal parameters changed to $\sigma = 1.2$ and $p = 1.0$. This shows that it pays off to incorporate the tonal optimisation into the mask optimisation process.

Parameter Selection of Electrostatic Halftoning

The principle idea of the electrostatic halftoning is that a specified number of particles (which correspond to the mask points) move around on a rectangular grid until a steady state is reached. This steady state is described by two main forces that act on the particles. On the one hand, there are repulsive forces between the particles which leads to homogeneous distribution of the particles in flat areas. On the other hand, there are attractive forces to the underlying force field. This force field is determined by the density function which forces particles to be more likely at locations with a larger density. If one has a closer look at the electrostatic halftoning method [168], one can see that there are two parameters which are worth of being studied in more detail:

First, there is the parameter α. In the electrostatic halftoning, the particles can freely move to any (continuous) position on the rectangular image domain. In the end, one is interested in particles lying on discrete grid locations. For this purpose, the parameter α allows to steer the amount of force which enforces the particle positions to be on the grid locations. The value for α is proposed to be 3.5, and is also adopted for the dithering within the analytic approach. However, after experimenting with this value, one can see that for the purpose of mask optimisation, it makes sense to choose a smaller value of α. An explanation for this fact is that the particles can then move more freely over the complete image domain, and are less likely to be stuck in a local minimum. In all the experiments on different images, the best value was $\alpha = 10^{-4}$, and can be fixed for future experiments. For the image trui, this means that the MSE is lowered from 35.68 to 32.46.

Another parameter is τ which represents the artificial time step size for the evolution of the particles. It inherently incorporates different constants such as the point charges and the Coulomb's constant. As in the original paper [168], a value of $\tau = 0.1$ is selected for the analytic approach so far. However, it is also a good idea to adapt this parameter. For example, choosing a value of $\tau = 1.5$ for the image trui yields a new MSE of 29.2. This also adapts the optimal presmoothing parameter σ to 1.0.

Optimising these two parameters along with the parameters σ and p gives a tremendous improvement of the mask quality. This could also be verified in other examples including different images.

Separation of Laplacian Sign

In the electrostatic halftoning, all particles are repelled by all others. This is also true for points lying across edges of the original image. Consequently, in the final mask, there is a minimum distance even between two adjacent mask points on either side of an edge. However, one can imagine that it is necessary to have a small or even no distance between two points such that they are able to recover very sharp edges. To remedy this issue, it might be a good idea to avoid the repelling forces across edges in the original image.

An important observation is that the sign of the Laplacian is different for locations across edges, and equal for locations on the same side of edges. Thus, the modification described above can be implemented by splitting the dithering into two parts. First, the analytic approach is applied for only the positive values of the Laplacian. To do so, Step 2 of Algorithm 5.4 is modified by computing the values of $\boldsymbol{a}$ for all pixels $i \in \Gamma$ as

$$a_i = \begin{cases} |(\boldsymbol{A}_N \boldsymbol{f}_\sigma)_i| & \text{if } (\boldsymbol{A}_N \boldsymbol{f}_\sigma)_i > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

Thereby only half of the desired mask points are used, i.e. one assumes the desired mask density to be $d/2$. Afterwards, in a second step, the electrostatic halftoning is accordingly performed for the negative values of the Laplacian, again by specifying only half of the desired final points. More precisely, the values of $\boldsymbol{a}$ in Step 2 are now given for all pixels $i \in \Gamma$ by

$$a_i = \begin{cases} |(\boldsymbol{A}_N \boldsymbol{f}_\sigma)_i| & \text{if } (\boldsymbol{A}_N \boldsymbol{f}_\sigma)_i < 0, \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

In total, the electrostatic halftoning is performed twice with half of the points for the two classes of the Laplacian. The two resulting two masks are then joined to obtain a final mask. More specifically, a mask point is set whenever this point is chosen in at least one of the two masks. As the two classes are disjoint, no mask point is selected in both masks. Thus, the desired mask density should be obtained. Note that the overall runtime of this modified version is smaller compared to the original analytic approach. Due to the fact that the complexity of the electrostatic halftoning grows quadratically with the number of points, it is more efficient to perform the dithering twice with half of the points. As a result of this separation, the MSE for the image trui can be reduced to 26.99. This is astonishing as this MSE is smaller than the MSE of 27.24 obtained after probabilistic sparsification and nonlocal pixel exchange. Moreover, the adapted analytic approach is extremely fast compared to the other approaches. This shows that a clever designed algorithm based on a theoretical framework with tuned parameters is able to outperform a greedy algorithm.

Improved Density Function

The analytic approach bases on the result in [23] that the density of the mask points should be chosen as an increasing function of the Laplacian magnitude. So far, this increasing function is assumed to be a power function with exponent p. A closer look into [23] reveals the following more precise relationship between the density μ and the Laplacian magnitude:

$$\frac{\mu^2}{|1 - \log(\mu)|} \approx c_0 |\Delta f|^2. \tag{5.4}$$

In contrast to this formula, a simplified version was employed before which neglects the denominator. However, it makes sense to include this information for obtaining a more accurate density function. After introducing a presmoothing and a power function as before, we thus assume the following discrete density:

$$\frac{\boldsymbol{\mu}^2}{|1 - \log(\boldsymbol{\mu})|} = c_0 |\boldsymbol{A}_N \boldsymbol{f}_\sigma|^{2p}. \tag{5.5}$$

Unfortunately, it is not possible to analytically solve this equation for $\boldsymbol{\mu}$. For given values of the Laplacian magnitude and a c_0, one can instead use a numerical approximation to evaluate $\boldsymbol{\mu}(c_0, |\boldsymbol{A}_N \boldsymbol{f}_\sigma|)$. For instance, a binary search algorithm can be applied due to the monotonicity of the function. This allows to compute the constant c_0 such that the resulting $\boldsymbol{\mu}$ becomes a probability density function. After tuning all the parameters, the final MSE for the image trui can be reduced to 26.04 (with $\sigma = 0.9$, $p = 1.07$, $\tau = 1.16$).

Adapted Distance Function

As a last improvement, one can adapt the distance function in electrostatic halftoning. In the original paper [168], the amount of force of a particles on each another decreases quadratically depending on the distance. More precisely, the force is scaled by $1/r^2$, with r being the distance between two points. The same decay is used for the influence of the force on the particles by the density field. However, this choice of the model design may not be the optimal one. To introduce more flexibility, one can thus assume a distance dependent decay of the forces given by $1/(r^q + \varepsilon)$. The small number $\varepsilon > 0$ is a small number which avoids singular values. In the experiments, it is fixed to $\varepsilon := 10^{-5}$. A smaller q leads to a slower decay of the mask points, and thus a more homogeneous behaviour over the whole image. A larger q would imply a faster decay depending on the distance, which leads to a localisation of the influences.

After incorporating also this final modification into the analytic approach, a final MSE of 25.71 is obtained for the image trui with only 4% of mask pixels. The values for the parameters are $\sigma = 1.0$, $p = 0.98$, $\tau = 1.55$ and $q = 2.51$. As one can see, a larger q and consequently a slightly more homogeneous distribution of the mask

Algorithm 5.5. Analytic Approach - Improved Version.

```
Input: Original image $\boldsymbol{f}$, Gaussian standard deviation $\sigma$, exponent $p$ for Laplacian,
  time step size $\tau$, exponent $q$ for distance function, desired pixel density $d$.
Compute:
    1. Perform Gaussian presmoothing with standard deviation $\sigma$: $\boldsymbol{f}_\sigma = K_\sigma * \boldsymbol{f}$.
    2. Optimise mask for positive Laplacian:
        i. Compute the Laplacian magnitude with positive sign, see Equation (5.2).
        ii. Determine the constant $c_{0,1}$ to obtain the probability density function $\boldsymbol{\mu}_1$,
            assuming a mask density of $d/2$.
        iii. Apply electrostatic halftoning on $\boldsymbol{\mu}_1$ with the parameters $\tau$ and $q$ to ob-
            tain $\boldsymbol{c}_1$.
    3. Optimise mask for negative Laplacian:
        i. Compute the Laplacian magnitude with negative sign, see Equation (5.3).
        ii. Determine the constant $c_{0,2}$ to obtain the probability density function $\boldsymbol{\mu}_2$,
            assuming a mask density of $d/2$.
        iii. Apply electrostatic halftoning on $\boldsymbol{\mu}_2$ with the parameters $\tau$ and $q$ to ob-
            tain $\boldsymbol{c}_2$.
    4. Obtain the final mask $\boldsymbol{c}$ of density $d$ by a logical ‘or’ operation on $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$.
Output: Optimised mask $\boldsymbol{c}$.
```

points is preferred. Interestingly, the best power p is set to 1, which confirms the result in [23].

Overview

As we have seen, there are a lot of modifications that help to qualitatively improve the analytic approach. For convenience, Algorithm 5.5 gives an overview of the final analytic approach. The individual parameters are determined using grid search to obtain an optimised mask. The final best mask and the corresponding tonal optimised inpainting for the image *trui* are depicted in Figure 5.7. Compared to the previous analytic approach in Figure 5.6 the reconstruction quality is much better. The increased tendency of the mask points towards the image edges seems to be the key factor for the improved performance. Interestingly, the final mask even outperforms the one obtained in a greedy way with the non-local pixel exchange. Although the analytic approach can not quite offer the same quality as the optimal control method, it has the clear advantage that the desired mask density can be chosen explicitly. However, the drawback of the method is that there are a lot of parameters which need to be optimised.

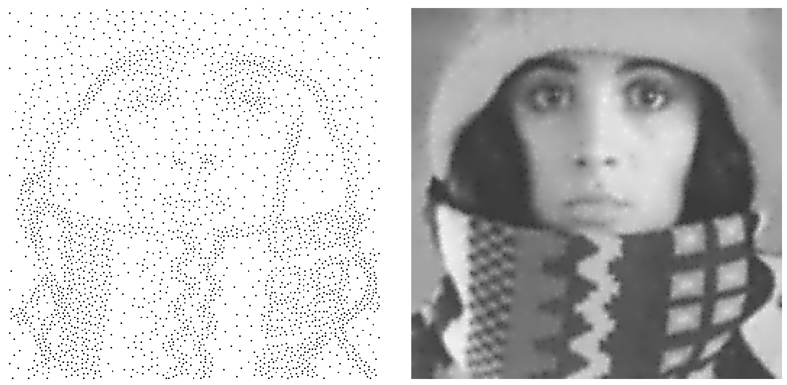

Figure 5.6. Spatial optimisation for harmonic inpainting with recent analytic approach. **Left:** Optimised mask (density: 4%) with Algorithm 5.4 (parameters: $\sigma = 1.6$, $p = 0.8$). **Right:** Corresponding tonal optimised inpainting (MSE: 42.00).

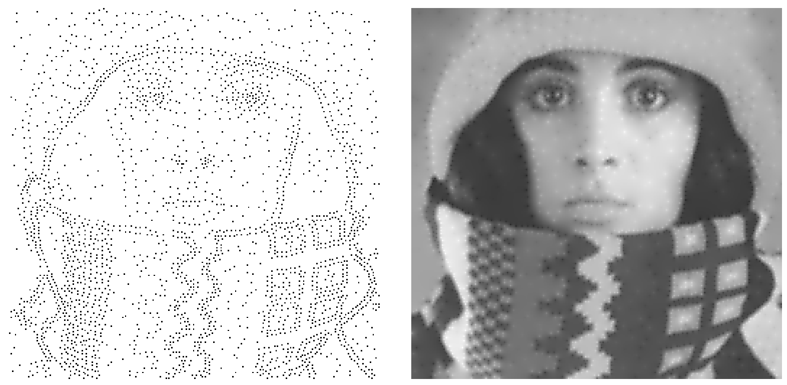

Figure 5.7. Spatial optimisation for harmonic inpainting with improved analytic approach. **Left:** Optimised mask (density: 4%) with Algorithm 5.5 (parameters: $\sigma = 1.0$, $p = 0.98$, $\tau = 1.55$, $q = 2.51$). **Right:** Corresponding tonal optimised inpainting (MSE: 25.71).

5.6 Spatial Optimisation with Discrete Green's Functions

As for the tonal optimisation, it is also possible to consider the problem of finding a mask from a different perspective, namely by using discrete Green's functions. It was discussed in Chapter 3 that the inpainting solution can be expressed in terms of discrete Green's functions. Moreover, it was shown in Chapter 4 that a tonal optimisation is equivalent to an optimisation of the coefficients of discrete Green's functions. Similarly, it is possible to reinterpret the problem of finding mask points as seeking the source points of the individual discrete Green's functions. The desired benefit of this strategy is that - in contrast to the probabilistic approaches - one has direct access to the influence of a single mask point on the inpainting result.

One can formulate the problem as follows. All discrete Green's functions on a rectangular domain given by the original image describe a dictionary. From this dictionary, one is interested in selecting a predefined number of Green's functions as atoms which allow to approximate the original image as good as possible. This approximation is given as the superposition of those Green's functions with corresponding coefficients (cf. Equation 3.55). As usual, these coefficients are supposed to be optimal for a given set of discrete Green's functions to evaluate the true quality of the resulting mask. More formally, we seek the source points $(x_1, y_1), \ldots, (x_L, y_L) \in \Gamma$, where L is the desired number of mask points, as well as the coefficients $a_1, \ldots, a_L \in \mathbb{R}$. These source points are equivalent to the final mask locations. It has already been shown in Section 4.6 that the optimal value of the constant c is given by the mean value of the original image and is independent of the chosen mask configuration. Thus, one does not have to care about this parameter. Additionally, the solvability condition (3.53) has to be fulfilled to obtain a valid inpainting solution.

Let us define the following energy:

$$E\left((x_k, y_k, a_k)_{k=1}^{L}\right) := \frac{1}{2}\left\|\sum_{k=1}^{L} a_k\ \boldsymbol{g}^{0}_{x_k,y_k} - \overline{\boldsymbol{f}}\right\|_2^2, \tag{5.6}$$

with $\overline{\boldsymbol{f}} := \boldsymbol{f} - \mu$ being the original image compensated by its mean value μ. Then, finding a mask can be formulated as the following optimisation problem.

$$\underset{\substack{(x_k,y_k)\in\Gamma,\, a_k\in\mathbb{R},\\ k=1,\ldots,L}}{\arg\min}\ E\left((x_k, y_k, a_k)_{k=1}^{L}\right). \tag{5.7}$$

After having found the best combination of coefficients and mask points, the final approximation $\boldsymbol{u}$ is given by

$$\boldsymbol{u} = \sum_{k=1}^{L} a_k\ \boldsymbol{g}^{0}_{x_k,y_k} + \mu. \tag{5.8}$$

In the literature, this problem is also known as sparse approximation problem [192, 195]. Here, the general task is to find a sparse subset of all atoms in a dictionary which leads to the best approximation of the original signal or image. A very popular method to find a minimiser is the so-called *orthogonal matching pursuit* [129]. In the following, the standard OMP approach is presented first. Then, it is demonstrated how one can improve the runtime performance of OMP with the help of an improved numerical scheme. Afterwards, an improved variant of OMP is proposed. Last but not least, a 3D optimisation approach relying on a gradient descent scheme is presented.

5.6.1 Orthogonal Matching Pursuit

Standard Formulation

Orthogonal matching pursuit (OMP) is a greedy algorithm which is well suited to find a sparse subset of atoms from a dictionary. Starting with an empty set, OMP successively selects a discrete Green's functions one after another by selecting the most relevant one in each step. In iteration step k, a new source point is inserted at that location where the corresponding Green's function correlates most strongly with the residuum. The residuum is given by $\boldsymbol{r}_k := \overline{\boldsymbol{f}} - \boldsymbol{u}_{k-1}$ between the original image and the up-to-date approximation $\boldsymbol{u}_{k-1}$. The initial inpainting $\boldsymbol{u}_0$ is set to zero everywhere. The k-th source point (x_k, y_k) is then determined as

$$\underset{(x,y)\in\Gamma}{\arg\max} \left| \frac{\left\langle \boldsymbol{r}_k, \boldsymbol{g}^0_{x,y} \right\rangle}{\left\langle \boldsymbol{g}^0_{x,y}, \boldsymbol{g}^0_{x,y} \right\rangle} \right| . \tag{5.9}$$

After adding a new Green's function in every step, the coefficients are optimised to obtain an updated best approximation $\boldsymbol{u}$ of the original image. Thereby, the solvability condition is not assumed in the intermediate steps to see the unveiled influence of the selected Green's functions. The condition is a restriction only on the coefficients altogether and is required to obtain a valid inpainting solution in the end. The intermediate optimisation of the coefficients has the following structure:

$$\underset{\boldsymbol{a}_k\in\mathbb{R}^k}{\arg\min} \left\| \boldsymbol{H}_k \boldsymbol{a}_k - \overline{\boldsymbol{f}} \right\|_2^2 , \tag{5.10}$$

Here, $\boldsymbol{H}_k \in \mathbb{R}^{|\Gamma|\times k}$ is the matrix containing the k already selected discrete Green's functions as columns, and $\boldsymbol{a}_k \in \mathbb{R}^k$ are the sought coefficients. Solving this problem can for instance be done analogous to the approach discussed in Section 4.6. The updated approximation is then obtained as $\boldsymbol{u}_k = \boldsymbol{H}_k \boldsymbol{a}_k$. In the end, the coefficients are optimised respecting the solvability condition to obtain a valid inpainting solution.

Note that in contrast to the probabilistic densification, it makes sense in OMP to perform an optimisation of the coefficients or equivalently the tonal data also within the intermediate steps. The reason for this lies in the selection of the next mask points. While the local error is considered in the densification, OMP considers the global correlation of the discrete Green's functions with the residuum. As a consequence, the decision in OMP bases on the overall distribution of the error and not on the local approximation quality.

Efficient Computation

Performing the OMP algorithm in a naïve way would require a lot of time. With the help of some optimisations, it is possible to lower the required runtime. First of all, one can precompute all the inner products $\left\langle \boldsymbol{g}^0_{x,y}, \boldsymbol{g}^0_{x,y} \right\rangle$ in (5.9) of all discrete Green's functions with themselves. In particular, as discussed in Section 3.3.4, these inner products can be computed very efficiently. The latter fact is also valid for the inner products of all discrete Green's functions with the residuum. Including these changes already leads to an immense reduction of the runtime.

Furthermore, one can accelerate the optimisation of the coefficients in each step by relying on the information from the previous step. As a basis, a QR decomposition is employed for solving the linear least-squares problem as suggested in [82]. Let $\boldsymbol{H} = \boldsymbol{QR}$ be the QR decomposition of a matrix $\boldsymbol{H} \in \mathbb{R}^{|\Gamma|\times L}$, with an orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{|\Gamma|\times|\Gamma|}$ and an upper triangular matrix $\boldsymbol{R} \in \mathbb{R}^{|\Gamma|\times k}$. Then, the solution of the least squares problem (5.10) is given by

$$\boldsymbol{a} = \boldsymbol{R}^{-1}\boldsymbol{Q}^\top\overline{\boldsymbol{f}}. \tag{5.11}$$

Note that the inversion of $\boldsymbol{R}$ is nothing else than a simple backwards substitution, making the computation of $\boldsymbol{a}$ very efficient once the QR decomposition is available. Especially for larger matrices, performing a QR decomposition is computationally expensive. Thus, instead of recomputing the QR decomposition in each step, a better idea is to use the information from the decomposition in the previous step. To see how this can be realised, a closer look at the QR decomposition algorithm is necessary.

Successful techniques for computing a QR decomposition of a matrix either rely on a modified Gram-Schmidt algorithm or on a Householder transform [194]. It has been noted in [82] that a Householder transform is numerically more robust and more efficient compared to a modified Gram-Schmidt procedure on the expense of being harder to implement. Thus, the Householder transform is employed here, and works as follows.

The basic idea is to construct an orthogonal matrix $\boldsymbol{Q}$ such that $\boldsymbol{Q}^\top\boldsymbol{H}$ results in an upper triangular matrix $\boldsymbol{R}$, i.e.

$$\boldsymbol{Q}^\top\boldsymbol{H} = \boldsymbol{R} \Leftrightarrow \boldsymbol{H} = \boldsymbol{QR}. \tag{5.12}$$

Thus, a QR decomposition is directly obtained after having found $\boldsymbol{Q}$. To do so, the columns of $\boldsymbol{H}$ are processed one after another from left to right. For each column $k \in \{1, \ldots, L\}$, a symmetric and orthogonal Householder matrix $\boldsymbol{M}_k$ is constructed, which has the form

$$\boldsymbol{M}_k := \boldsymbol{I} - \tau_k\, \boldsymbol{v}_k\, \boldsymbol{v}_k^\top, \tag{5.13}$$

with the identity matrix $\boldsymbol{I} \in \mathbb{R}^{|\Gamma|\times|\Gamma|}$, and unknown $\tau_k \in \mathbb{R}$, $\boldsymbol{v}_k \in \mathbb{R}^{|\Gamma|}$. The structure of $\boldsymbol{v}_k$ is as follows:

$$\boldsymbol{v}_k = \big(0, \ldots, 0, 1, v_{k+1}, \ldots, v_{|\Gamma|}\big)^\top. \tag{5.14}$$

The values of τ_k and the unknown values of $\boldsymbol{v}_k$ are determined such that the following property is fulfilled: For an arbitrary vector $\boldsymbol{b} \in \mathbb{R}^{|\Gamma|}$, the result of $\boldsymbol{M}_k\boldsymbol{b}$ shall be identical to zero for all entries with an index larger than k. All other entries remain unmodified by construction. For more information on how the values of $\boldsymbol{v}$ can be determined, please see [194]. Consecutively applying the Householder matrices $\boldsymbol{M}_1, \dots, \boldsymbol{M}_L$ thus allows to bring all values of the lower triangular part to zero from left to right. For instance, applying the k-th Householder transform leads to the following outcome:

$$\boldsymbol{M}_k \begin{pmatrix} a_{1,1} & \cdots & \cdots & \cdots & \cdots & a_{1,L} \\ 0 & \ddots & & & & \vdots \\ \vdots & 0 & a_{k,k} & a_{k,k+1} & \cdots & a_{k,L} \\ \vdots & \vdots & a_{k+1,k} & a_{k+1,k+1} & & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & a_{L,k} & a_{L,k+1} & \cdots & a_{L,L} \end{pmatrix} = \begin{pmatrix} a_{1,1} & \cdots & \cdots & \cdots & a_{1,L} \\ 0 & \ddots & & & \vdots \\ \vdots & & \star & \star \cdots & \star \\ \vdots & & 0 & \star & \vdots \\ \vdots & & \vdots & \vdots & \ddots \\ 0 & \cdots & 0 & \star \cdots & \star \end{pmatrix} \tag{5.15}$$

The entries $a_{i,j}$ with $i, j \geq k$ might get modified, while the rest remains unchanged. The matrix $\boldsymbol{Q}^\top$ is finally given by

$$\boldsymbol{Q}^\top = \boldsymbol{M}_L \cdot \ldots \cdot \boldsymbol{M}_1. \tag{5.16}$$

As a result, $\boldsymbol{Q}^\top\boldsymbol{H}$ represents a triangular matrix $\boldsymbol{R}$. The orthogonality of the matrix $\boldsymbol{Q} = \boldsymbol{M}_1 \cdot \ldots \cdot \boldsymbol{M}_L$ is verified easily from the orthogonality of the individual Householder matrices. Thus, the QR decomposition is complete, and the solution of the least squares problem can be computed according to (5.11). Note that this only requires to have the transpose of the matrix $\boldsymbol{Q}$, so there is no need to compute $\boldsymbol{Q}$.

One advantage of the Householder transform to obtain the QR decomposition is that the large matrix $\boldsymbol{Q}$ respectively $\boldsymbol{Q}^\top$ does not have to be stored explicitly. In contrast, all the necessary information for recovering these matrices is given by $\boldsymbol{M}_1, \dots, \boldsymbol{M}_L$ which in turn are represented by the values of $\tau_1, \dots, \tau_L$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_L$. All relevant operations can be performed from this data. In practice, one can even store the complete information of $\boldsymbol{Q}$ in the free memory of $\boldsymbol{R}$: The relevant information of the vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_L$ fit on the lower diagonal, while $\tau_1, \dots, \tau_L$ can be written on the main diagonal of $\boldsymbol{R}$. If it is not required to keep the matrix $\boldsymbol{H}$, one can thus store its QR decomposition in the same memory by replacing the columns one after another.

Let us now consider how one can reuse the QR decomposition within the OMP algorithm. To this end, let a system matrix $\boldsymbol{H}_{k-1} \in \mathbb{R}^{|\Gamma| \times (k-1)}$ be given after having selected $k-1$ discrete Green's functions. Additionally, the QR decomposition $\boldsymbol{H}_{k-1} = \boldsymbol{Q}_{k-1}\boldsymbol{R}_{k-1}$ is assumed, and let $\boldsymbol{b}_{k-1} := \boldsymbol{Q}_{k-1}^\top \overline{\boldsymbol{f}}$ be computed, such that the current coefficients are given by the backwards substitution $\boldsymbol{a}_{k-1} = \boldsymbol{R}_{k-1}\boldsymbol{b}_{k-1}$ according to Equation (5.11). Selecting a new Green's function $\boldsymbol{g}^0_{x_k,y_k}$ in step k means that the system matrix $\boldsymbol{H}_{k-1}$ is extended by one more column. This yields a new matrix $\boldsymbol{H}_k \in \mathbb{R}^{|\Gamma| \times k}$ by appending the additional column at the end:

$$\boldsymbol{H}_k = \left(\boldsymbol{H}_{k-1} \mid \boldsymbol{g}^0_{x_k,y_k}\right) . \tag{5.17}$$

Applying $\boldsymbol{Q}^\top_{k-1}$ from the existing QR decomposition yields

$$\boldsymbol{Q}^\top_{k-1}\boldsymbol{H}_k = \left(\boldsymbol{Q}^\top_{k-1}\boldsymbol{H}_{k-1} \mid \boldsymbol{Q}^\top_{k-1}\boldsymbol{g}^0_{x_k,y_k}\right) = \left(\boldsymbol{R}_{k-1} \mid \boldsymbol{Q}^\top_{k-1}\boldsymbol{g}^0_{x_k,y_k}\right) . \tag{5.18}$$

In order to obtain a valid QR decomposition of $\boldsymbol{H}_k$, one has to transform this matrix into an upper triangular matrix again. To achieve this, all one has to do is to compute a Householder transform $\boldsymbol{M}_k$ for the last column. This yields the updated $\boldsymbol{Q}^\top_k = \boldsymbol{M}_k\boldsymbol{Q}^\top_{k-1}$ and corresponding diagonal matrix

$$\boldsymbol{R}_k = \left(\boldsymbol{R}_{k-1} \mid \boldsymbol{Q}^\top_k\boldsymbol{g}^0_{x_k,y_k}\right) . \tag{5.19}$$

This in turn allows to update $\boldsymbol{b}_k = \boldsymbol{M}_k\boldsymbol{b}_{k-1}$, the coefficients $\boldsymbol{a}_k = \boldsymbol{R}^{-1}_k\boldsymbol{b}_k$ as well as the approximation $\boldsymbol{u}_k = \boldsymbol{H}_k\boldsymbol{a}_k$. A detailed description of the complete OMP method including all optimisations is shown in Algorithm 5.6.

Improved Variant

As we have seen, the standard OMP approach is not optimal. In order to improve the result, the following conceptual issue can be made out. The selection of the mask points and the corresponding coefficients are found without any interaction in each step: First, a location is found, while the coefficients are optimised in a second step. In contrast, a selection based on the joint information would be better. To remedy this problem, we propose a modified orthogonal matching pursuit. The idea is to select the Green's function which allows for the highest improvement in the approximation. This enables a coupling between positions and coefficients in each step. More formally, instead of (5.9), we are looking for

$$\operatorname*{arg\,min}_{(x,y)\in\Gamma} \left\{\min_{\alpha} \left\|\boldsymbol{r}_k - \alpha\, \boldsymbol{g}^0_{x,y}\right\|^2_2\right\} . \tag{5.20}$$

Interestingly, the inner minimisation problem can be computed explicitly, which allows to simplify this best point selection to

$$\operatorname*{arg\,max}_{(x,y)\in\Gamma} \left|\frac{\left\langle \boldsymbol{r}_k, \boldsymbol{g}^0_{x,y}\right\rangle^2}{\left\langle \boldsymbol{g}^0_{x,y}, \boldsymbol{g}^0_{x,y}\right\rangle}\right| . \tag{5.21}$$

Compared to (5.9), the only difference is the square in the nominator. Thus, these two approaches are of the same complexity. Besides this small change in Step (c) in Algorithm 5.6, all other steps of the OMP remain the same.

```
Algorithm 5.6. Orthogonal Matching Pursuit (OMP)
Input: Image $\boldsymbol{f}$, desired number $L$ of mask points.
Initialisation:
    1. Initialise $\boldsymbol{u}_0 := \boldsymbol{0}$, compensated image $\overline{\boldsymbol{f}} := \boldsymbol{f} - \mu$ by its mean, $\boldsymbol{b}_0 = \overline{\boldsymbol{f}}$, empty
       matrices $\boldsymbol{H}_0, \boldsymbol{R}_0$, and $\boldsymbol{Q}_0 = \boldsymbol{I} \in \mathbb{R}^{|\Gamma|\times|\Gamma|}$.
    2. Precompute inner products $\left\langle \boldsymbol{g}^0_{x,y}, \boldsymbol{g}^0_{x,y} \right\rangle$ for all $(x,y) \in \Gamma$, see Section 3.3.4.
Compute:
    1. Do for all $k = 1, \dots, L$:
        (a) Compute residuum $\boldsymbol{r}_k := \overline{\boldsymbol{f}} - \boldsymbol{u}_k$.
        (b) Employ Algorithm 3.1 to get inner products of all Green's functions
            with $\boldsymbol{r}_k$.
        (c) Find best new pixel location $(x_k, y_k)$ with (5.9).
        (d) Append the discrete Green's function $\boldsymbol{g}^0_{x_k,y_k}$ to system matrix and obtain
            $\boldsymbol{H}_k$, see (5.17).
        (e) Obtain new coefficients using updated QR decomposition of $\boldsymbol{H}_k \in \mathbb{R}^{|\Gamma|\times k}$:
            (i) Obtain Householder transform $\boldsymbol{M}_k$ based on $\boldsymbol{Q}^\top_{k-1}\boldsymbol{g}^0_{x_k,y_k}$.
            (ii) With $\boldsymbol{Q}^\top_k = \boldsymbol{M}_k\boldsymbol{Q}^\top_{k-1}$, obtain $\boldsymbol{R}_k$ according to (5.19).
            (iii) Update $\boldsymbol{b}_k = \boldsymbol{M}_k\boldsymbol{b}_{k-1}$ and coefficients $\boldsymbol{a}_k = \boldsymbol{R}^{-1}_k\boldsymbol{b}_k$.
        (f) Update approximation $\boldsymbol{u}_k = \boldsymbol{H}_k\boldsymbol{a}_k$.
    2. Perform a tonal optimisation with all selected mask points and enabled solv-
       ability condition to obtain final inpainting $\boldsymbol{u}$, see Algorithm 4.5.
Output: Optimised mask locations $(x_k, y_k)$, $k = 1, \dots, L$ and corresponding tonal
    optimised inpainting $\boldsymbol{u}$.
```

Example Results

For the image *trui*, the results of both the standard OMP approach and its improved variant can be found in Figure 5.8 and Figure 5.9. Again, the target mask density was 4%. As expected, the improved variant clearly yields better masks than the standard version of OMP. In contrast, the fact that both approaches yield comparable qualities for harmonic and biharmonic inpainting is rather surprising. Unfortunately, the reconstructions are not very convincing compared to previous approaches.

The mask points show an obvious tendency towards the centre of the image, and there is another large amount of points along the image boundaries. The former aspect can be explained from the fact that discrete Green's functions with source points close to the image centre have a larger region of influence compared to points

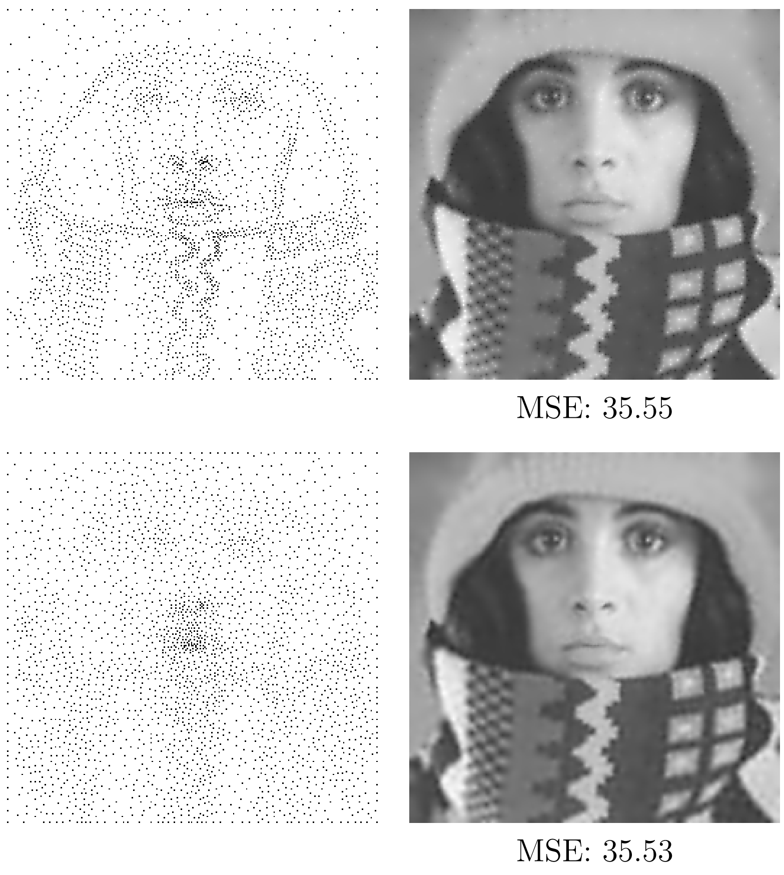

Figure 5.8. Spatial optimisation with standard OMP. **Top row:** Harmonic inpainting. **Bottom row:** Biharmonic inpainting. **Left column:** Optimised masks (density: 4%). **Right column:** Corresponding tonal optimised inpaintings.

further away. Thus, more points are selected at the centre especially in earlier stages. As a consequence, the errors within the regions further away tend to become larger at some point. This results in boundary points being added because the corresponding discrete Green's functions are able to cover those regions best.

5.6.2 Gradient Descent

As for the stochastic approaches, it is clear that OMP and also its improved variant cannot find the optimal mask. The fact that the points are inserted one after another does not allow for a globally best result: Once a mask point is selected, it will be kept forever. One can imagine that the mask points would benefit from a subsequent rearrangement of their configuration to allow for an adaption to the local image structure. This is also the idea behind the nonlocal pixel exchange. While this approach was primarily a brute force algorithm, discrete Green's functions allow a much more elaborated global optimisation strategy.

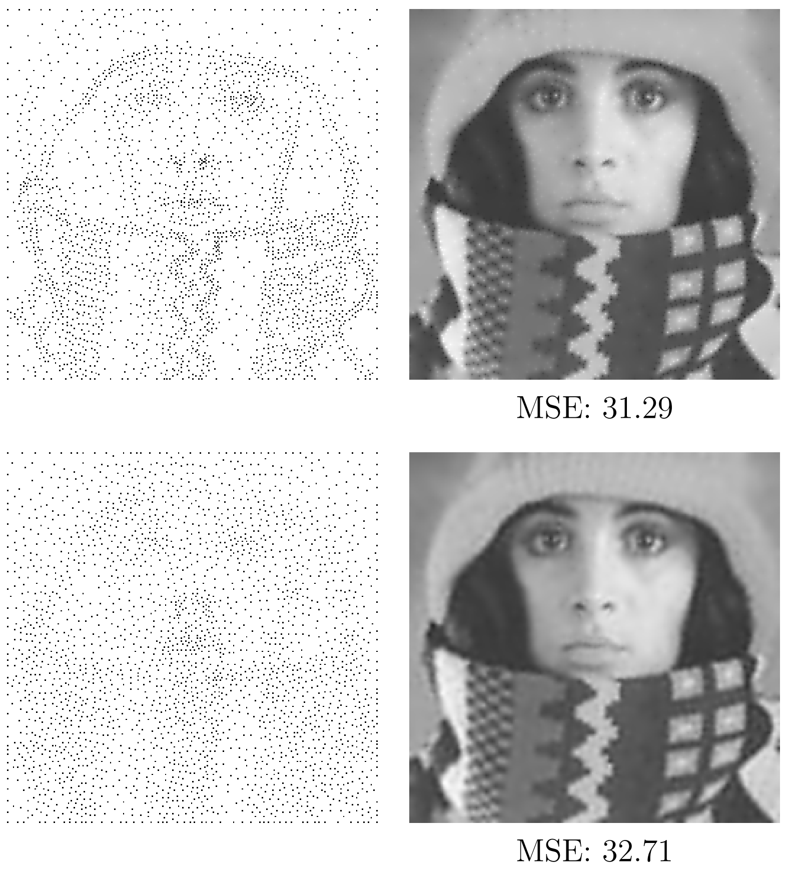

Figure 5.9. Spatial optimisation with improved variant of OMP. **Top row:** Harmonic inpainting. **Bottom row:** Biharmonic inpainting. **Left column:** Optimised masks (density: 4%). **Right column:** Corresponding tonal optimised inpaintings.

Assume we are given some mask which has been found by any previous optimisation strategy, such as OMP for example. Then, it should be possible for the mask points to move around within the spatial domain, while at the same time allowing to adapt their coefficients, in order to improve the inpainting result. The hope is to get closer to a global optimum of the energy in (5.7). This procedure can be considered as a 3D optimisation strategy.

A gradient descent scheme helps to realise this abstract formulation, and works as follows. Given an initial configuration $\boldsymbol{w}^0 := (x_k^0, y_k^0, a_k^0)_{k=1}^L$ of the mask points and corresponding coefficients, we obtain the following general update scheme for $n \geq 0$:

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n - \alpha \boldsymbol{\nabla} E(\boldsymbol{w}^n), \tag{5.22}$$

with a step size $\alpha \in \mathbb{R}^+$. It is clear that during this evolution, the entries of $\boldsymbol{w}$ can become real valued. In other words, the mask points do not necessarily have to lie at the discrete grid locations Γ. However, in contrast to circumventing this issue by a projection step after each iteration, we allow this degree of freedom. In particular, as the coefficients are also continuous, the mask points should possess the same liberty. This can be seen as a relaxation of the problem. During the evolution, we only make sure that no mask point leaves the image domain by a back projection. To asses a discrete Green's function $\boldsymbol{g}^0_{x,y}$ at a continuous source point (x, y), we use a bilinear interpolation of the discrete Green's functions at the four surrounding grid points. This procedure has the advantage that the Green's function $\boldsymbol{g}^0_{x,y}$ coincides with the usual discrete Green's function whenever (x, y) lies at a discrete grid location. This means that these Green's functions allow to retrieve an actual inpainting solution whenever all mask points are located at grid locations. The computation of the bilinear interpolation is not very time consuming due to the efficient construction strategies discussed in Section 3.3.5. Concerning the gradient of E, it is possible to write down an explicit formulation of the derivative with respect to a coefficient a_k:

$$\frac{\partial E}{\partial a_k}(\boldsymbol{w}^n) = \left\langle \boldsymbol{g}^0_{x_k^n, y_k^n},\ \boldsymbol{u}^n - \overline{\boldsymbol{f}} \right\rangle . \tag{5.23}$$

Here, $\boldsymbol{u}^n = \sum_{k=1}^{L} a_k^n \, \boldsymbol{g}^0_{x_k^n, y_k^n}$ is the current approximation at step n. Unfortunately, the spatial derivatives cannot be derived as easy due to the nonlinear dependency of the inpainting solution $\boldsymbol{u}$ on the mask locations. An important observation is that when a single mask point moves to another location while leaving its coefficient unmodified, it is easy to update the inpainting. All one has to do is remove the Green's function at the old position, and add a Green's function at the new position. In particular, it is not necessary to perform a complete inpainting. This allows to easily approximate the effect of a small change in the mask location. For this reason, we can approximate the inner derivatives with the help of finite differences, similar to the tonal optimisation for nonlinear inpainting operators in Section 4.8. This leads to the following formulation of the gradient at some location $(x_k, y_k) \in \Gamma$:

$$\frac{\partial E}{\partial x_k}(\boldsymbol{w}^n) \approx \left\langle a_k^n \cdot \frac{\boldsymbol{g}^0_{x_k^n+\eta, y_k^n} - \boldsymbol{g}^0_{x_k^n, y_k^n}}{\eta},\ \boldsymbol{u}^n - \overline{\boldsymbol{f}} \right\rangle , \tag{5.24}$$

$$\frac{\partial E}{\partial y_k}(\boldsymbol{w}^n) \approx \left\langle a_k^n \cdot \frac{\boldsymbol{g}^0_{x_k^n, y_k^n+\eta} - \boldsymbol{g}^0_{x_k^n, y_k^n}}{\eta},\ \boldsymbol{u}^{\boldsymbol{n}} - \overline{\boldsymbol{f}} \right\rangle , \tag{5.25}$$

with $\eta \in \mathbb{R}^+$ representing a small perturbation in the location. Thus, it is possible to obtain the gradient by a simple formula involving only a few discrete Green's functions. This is a major difference to the conventional inpainting formulation where the computation of the gradient would only be possible by performing several inpaintings. Unfortunately, this would suffer from a large runtime. The discrete Green's functions on the other hand lead to a huge performance gain. Moreover, the standard formulation would not support continuous mask locations.

An important aspect is that the overall goal is to find discrete mask locations. For this reason, it is not desired to obtain a continuous mask where several points will be mapped to the very same discrete grid point. To prevent this from happening, the following countermeasure is performed after each gradient descent step. If two points would potentially be mapped to the same discrete point, one of them is relocated to another grid position with large reconstruction error. This allows to improve the quality in those regions while having no drawback from removing a practically superfluous point. Furthermore, the following procedure allows to further improve the final mask. After a certain amount of iterations, the coefficients of the discrete Green's functions are considered. Whenever the value is close to zero, the corresponding mask point obviously has a negligible contribution on the inpainting result. Thus, one can easily relocate those points to regions of higher error, too.

Figure 5.10 shows an example result of the gradient descent approach. Starting with an initial mask obtained by OMP with a density of 4%, the method yields very good results for the harmonic inpainting operator. With discrete mask locations, the MSE of the reconstruction is 25.6. This is a very good result, only the optimal control approach is able to yield a better mask. Keeping the continuous mask locations the reconstruction with bilinear interpolated discrete Green's functions even has a final MSE of 16.81. This is much better than any method before could achieve for harmonic inpainting. Since continuous tonal values are allowed, it is hard to argue why continuous mask locations should not be permitted.

Considering the parameters, $\eta = 0.1$ was found to be a good value for computing the finite differences. The step size of the gradient descent step has to be chosen very small to obtain a stable process. Even for the harmonic operator, the maximum value was $\alpha = 10^{-5}$, leading to a very long runtime. Interestingly, experiments showed that choosing a different step size for the spatial and tonal direction improves the final mask. In fact, a smaller value for the tonal direction typically gave better results. This can be explained with the help of the previous observation in Section 5.3: If the tonal values are optimised too early the mask points only have little motivation to move to another location. Thus, a slower adaptation of the coefficients and consequently of the tonal data enables a better spatial optimisation. Selecting a step size of 10^{-5} for the tonal direction and a step size of around 1 in spatial direction leads to a stable process. The method converges after performing approximately 1600 gradient descent steps for all points. Unfortunately, for the biharmonic operator, the algorithm diverges even with double precision no matter how small the step sizes are chosen. Apparently, the very high condition number of the system matrix in (5.7) leads to an extremely hard numerical problem (cf. Table 4.2 in Section 4.6).

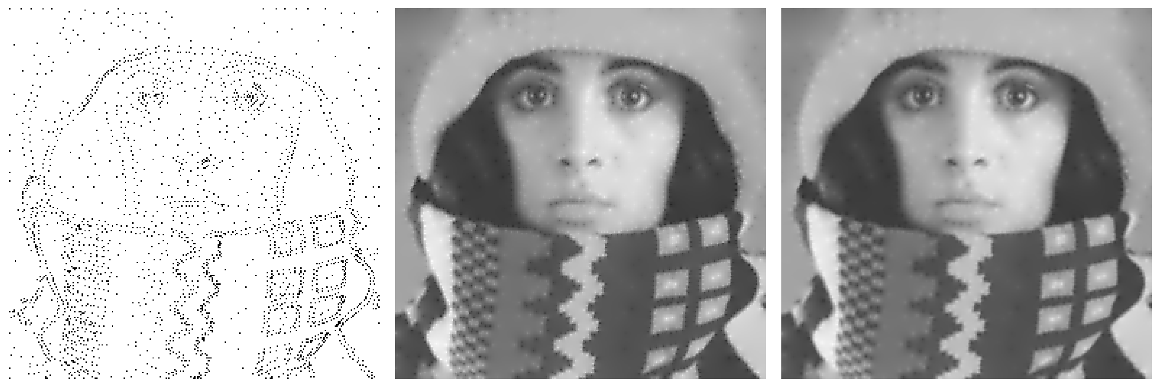

Figure 5.10. Spatial optimisation for homogeneous diffusion inpainting with gradient descent using discrete Green's functions and a mask density of 4%. **Left:** Final mask with points at grid locations. **Centre:** Corresponding tonal optimised inpainting (MSE: 25.6). **Right:** Optimised inpainting with continuous mask locations (MSE: 16.81).

5.7 Finding Masks for Nonlinear Inpainting Operators

It is clear that the promising inpainting performance of EED gives rise to find optimised masks also for nonlinear inpainting operators. Unfortunately, most of the the discussed methods are not applicable for performing a spatial optimisation for nonlinear operators. The reasons for this are manifold. When it comes to the primal-dual approach, one could try to incorporate the dependency of the nonlinear operator on the inpainting by approximating the inner derivatives with finite differences. However, besides the much larger runtime, it is not obvious if the minimisation of the highly non-convex energy would lead to a good local minimum. When it comes to the discrete Green's functions, they have only been considered for linear operators and it is not easy to extend the theory to nonlinear operators. Also the analytic approach of Belhachmi *et al.* only offers a theory for the harmonic operator. So far, there exist no results for other operators. Those topics remain an open question for future work.

Luckily, the probabilistic approaches to not explicitly rely on any structure of the inpainting operator. In contrast, the inpainting procedure can be seen as a black box in these methods which can potentially be replaced by any other inpainting method. Even though the probabilistic densification and sparsification as well as the nonlocal pixel exchange did not yield the best masks, they include reliable procedures which are applicable even in such complex scenarios.

In an experiment, a mask with a target density of 4% is supposed to be found for EED inpainting. As usual, the the image *trui* represents the original image. To this end, the parameters of the probabilistic sparsification and the subsequent nonlocal pixel exchange are tuned to obtain the best reconstruction. The optimal values are given by $p = 0.05$, $q = 10^{-6}$ and $m = 30$. They lead to the final mask which has already been shown in Figure 4.8 of Section 4.8. It achieves a remarkable MSE of 12.62. As shown in that previous Section, it is even possible to obtain a MSE of 10.79 after applying the tonal optimisation designed for nonlinear operators.

To the best of my knowledge, such a reconstruction quality has never been achieved before with PDE-based inpainting and a mask density of only 4% for the image *trui*. In particular, one can come to the conclusion that EED inpainting outperforms the linear counterparts when it comes to the reconstruction quality. However, one also has to keep in mind the enormous amount of time needed for performing the spatial and tonal optimisation with EED. For this reason, linear operators might be better suited for practical applications. In particular, note that the MSE of 16.14 obtained with optimal control and the biharmonic operator or the MSE of 16.81 with the harmonic operator and continuous mask locations are even very close to the EED result. Thus, it would be fatal to ignore the linear operators when designing compression codecs.

5.8 Conclusion

In this chapter, we have considered different methods for obtaining optimised mask locations for PDE-based inpainting. Similar to the tonal optimisation problem, there is no ultimate best method for the spatial optimisation as well. We have considered different approaches which can be used depending on the specifications such as the available runtime, the used inpainting operator, the hardware or the ability to specify the mask density. As a result, it is possible to obtain remarkable reconstructions, even with the conceptually simple linear inpainting operators such as the Laplacian or the biharmonic operator.

To obtain the best possible masks for the harmonic or biharmonic operators, the optimal control approach gives the best results. However, it has the clear disadvantage that it is not possible to specify the mask density explicitly. Finding a parameter which yields the desired number of points can thus be time-consuming. Moreover, it is necessary to perform the computations on a parallel hardware such as a GPU to reduce the runtime to a feasible amount.

The probabilistic approaches clearly have the advantage that the mask density can be specified. Unfortunately, the final masks are not the optimal ones and the runtime can be large. Nevertheless, the probabilistic approaches are straightforward to implement and very generic: They are the only approaches which are suited for any inpainting operator, in particular for nonlinear ones. Together with the tonal optimisation strategy for nonlinear operators from Section 4.8, this allowed to come up with a reconstruction of hitherto unparalleled quality with EED inpainting.

The analytic approach relying on the continuous theory of Belhachmi *et al.* is a promising alternative approach for harmonic inpainting. Here, the density can be specified explicitly. Depending on the used halftoning, the method can be very fast and even real-time capable. However, to obtain competitive results, a more sophisticated halftoning approach is required. Even with a GPU implementation, this leads to a higher runtime, which is also caused by the need to select the various parameters. Of course, an extension of the results of Belhachmi *et al.* to the biharmonic or even nonlinear operators are a topic for future work.

Both methods relying on the theory of discrete Green's functions allow to specify the desired density, too. Even though OMP has the advantage of being completely parameter-free and relatively fast, even the improved variant could not yield quite as good results as all other methods. However, the gradient descent strategy is able to improve this result, and to find a better mask than the analytic or probabilistic approaches for the harmonic operator. Unlike all other methods, it can even optimise for continuous mask locations, leading to the best results when no restriction to discrete grid locations are required. Unfortunately, no optimisation for the biharmonic operator could be performed due to the numerically demanding problem.

An additional result of this chapter is that an early tonal optimisation can sometimes hinder a spatial optimisation: The tonal optimisation tries to compensate for imperfect mask locations which reduces their desire to move to other locations. From another perspective, one can also say that the risk of being stuck in a bad local minimum is much higher due to a reduced smoothing effect.

Chapter 6

PDE-based Image Compression

In this chapter, we will focus on actual image compression applications. In the previous chapters, we have discussed various optimisation approaches for selecting the best inpainting data for different scenarios. These methods will serve as building blocks which we will use for designing three different compression algorithms in the following. Thereby, we want to find out the potential of linear inpainting operators in a compression context. As we have seen, even these conceptually simple operators have a very promising reconstruction capability and allow for algorithmic benefits. This gives rise to the hope that we can obtain well-performing compression methods with these operators. In particular, our goal is to come up with codecs which are able to compete with standard approaches such as JPEG, JPEG2000 or HEVC.

As described in Section 1.2.2, the optimal data might lead to good reconstructions, but can be hard to stored efficiently. Thus, we will have to face the problem of finding a good trade-off between data optimality and coding costs: While the reconstruction quality was the only criterion to judge the individual optimisation strategies so far, we now have to judge the selected data by taking into account the final file size as well.

In the following, we will first consider a dedicated compression algorithm for depth maps in Section 6.1. As we will see, an adapted linear inpainting is perfectly suited for these specific images. Afterwards in Section 6.2, we will concentrate on general images. Here, we will use the best possible inpainting data to obtain a competitive image compression algorithm. While the first two compression methods mainly focus on qualitative aspects, the design of the third compression framework in Section 6.3 will pay attention to both quality and decoding speed.

6.1 Compression of Depth Maps

In recent years, 3D cinema technology has become increasingly popular. In the corresponding so called multi-view video + depth (MVD) format, multiple images are captured from different perspectives along with their respective depth images. To cope with the huge amount of data, an efficient compression is indispensable. Combined compression methods have been presented (see e.g. [163]), where the correlated information between the colour images and the corresponding depth maps is exploited. It is also possible to incorporate a temporal component into the compression framework. In the following, however, we focus exclusively on the problem of depth map compression.

In [125] the authors have presented a related method which encodes the grey or colour values on both sides of image edges. In contrast to nonlinear anisotropic diffusion processes, a basic homogeneous diffusion inpainting is sufficient to reconstruct the image. For cartoon-like images the method could outperform JPEG2000. Indeed, for depth map compression, extracting and storing edges is actually a natural idea as they are crucial to obtain a good visual impression of the image. However, due to the fact that the above codec cannot handle homogeneous variations, it turns out that its application to depth maps leads to unsatisfactory results.

In [81, 119] modified versions of this edge-based approach have been suggested. The main changes consist of adding grey values at regular mask points, exploiting different edge detectors, and encoding parts of the extracted data with other methods. Similarly, homogeneous diffusion can be incorporated in existing block-based approaches where additional edge information is used to attain sharp edges [50]. However, all these methods are either data intensive or lead to a fairly complex overall codec.

Other approaches try to split the depth image recursively into smaller parts, resulting in a tree structure, and recover the depth map on the lowest tree level by means of linear interpolation [140, 201]. In [99], the depth map is approximated by linear functions within segments. A similar method, also working on segments, has been introduced in [218] where mainly bilinear interpolation of data on a regular grid has been used to reconstruct the depth information. All these methods have the drawback that a lot of information has to be stored to be sufficiently flexible.

In the following, the aforementioned problems are addressed by presenting a novel approach which has partly been published in [2]. While it is also based on homogeneous diffusion inpainting, it differs from [125, 81, 119] by the fact that it replaces edges by closed contours that result from a segmentation. This creates a decoupling into sub-problems and allows to benefit from parallel implementations. More importantly, by assuming homogeneous Neumann boundary conditions between segments, it is unnecessary to store grey values at contours. The corresponding grey values to be stored are additionally optimised. In the end, we do not only achieve a codec for depth maps that outperforms JPEG [150] and JPEG2000 [191], but even has the potential to compete with the substantially more complex HEVC (High Efficiency Video Coding), which is one of the most favourable methods to encode this type of images [145]. Based on first results in [157], the complete approach has been published in 2013 at the 4th International Conference on Scale Space and Variational Methods in Computer Vision [2].

The following sections are organised as follows. First, we introduce segment-based homogeneous diffusion in Section 6.1.1. Based on this concept we describe our encoding process in Section 6.1.2 and discuss the corresponding decoding steps in Section 6.1.3. A discussion on the parameter selection is given in Section 6.1.4, while an extension to the biharmonic operator is the topic of Section 6.1.5. A comparison to standard approaches will be presented in Section 6.1.6. A summary in Section 6.1.7 gives a conclusion and shows potential future work.

6.1.1 Segment-based Homogeneous Diffusion (SBHD)

Depth maps can be characterised by their piecewise smoothness. While PDE-based inpainting methods with linear inpainting operators are perfectly suited to reconstruct smooth areas, life becomes harder when it comes to recovering sharp edges. As done in [125, 81, 119], one can address this issue by selecting mask points around the contours which makes sense according to the theory of Belhachmi *et al.*. However, the drawback is that a lot of additional information has to be stored only for the edges. Thus, a natural idea is to explicitly store the relevant edges and use this information within the inpainting process. By doing so, the mask points are mainly supposed to describe smooth transitions in the image and hence to not have to be selected around sharp edges anymore. To realise this strategy, the new so-called *segment-based homogeneous diffusion (SBHD)* will be introduced in the following. It represents the central feature of the novel compression approach.

SBHD relies on a segmentation of the image domain Ω into several sub-domains. For each of the segments, we again assume the tonal values of an image $f : \Omega \to \mathbb{R}$ to be given at the mask points in $\Omega_K \subset \Omega$. This information is then used to fill in the respective segment. More precisely, the inpainting u is obtained by solving

$$-\Delta u = 0, \tag{6.1}$$

subject to the following *mixed boundary conditions*:

$$\begin{aligned} u &= f && \text{at mask points} && (\textit{Dirichlet boundary conditions}), \\ \partial_{\boldsymbol{n}} u &= 0 && \text{at segment boundaries} && (\textit{homogeneous Neumann boundary conditions}). \end{aligned}$$

Thereby, $\boldsymbol{n}$ is the unit normal vector to the respective segment boundary, and $\partial_{\boldsymbol{n}} u$ denotes the partial derivative of u in this direction. As in Section 2.3, the discretisation of this partial differential equation can be realised in a straightforward way using finite differences [139]. Then, as long as there is at least one mask pixel in each segment, there exists an unique solution of the discrete problem (cf. [4]).

As a result of SBHD we obtain an image containing (i) sharp edges at segment boundaries and (ii) smooth transitions within segments steered by the values at the mask points. These transitions can also represent unsharp edges. SBHD is therefore well suited for the representation of depth images for example.

An important advantage of this SBHD is the fact that, by construction, each mask point does not have a global influence: Its impact is limited by the respective segment boundaries. This allows a segment-wise parallel processing. In order to get a solution of the diffusion equation we make use of the *fast explicit diffusion (FED)* scheme [87] together with a CUDA implementation on the GPU. Similar to the tonal optimisation experiments in Section 4.7, one can observe a substantial speedup compared to a pure CPU implementation due to the parallelism. Another speedup is gained by reducing the number of required iterations to reach a steady state. This is achieved with a good initialisation of the inpainting solution: All unknown values at non-mask points are initially set to the mean value of the known grey values within the individual segments. In particular for segments with only one mask point this means that the inpainting solution is directly obtained.

6.1.2 Data Extraction and Encoding

Given some segmentation as well as interpolation data, SBHD is able to recover an image. Similar to the standard inpainting approaches the question is how the data can be chosen to obtain the best inpainting. In the following, this problem is split into three sub-problems. First, we seek for a good segmentation which allows for a good inpainting with SBHD. Given this fixed segmentation, the mask points are selected afterwards. Finally, the corresponding tonal values are optimised. The advantage of this separate optimisation procedure is that the overall complexity of the problem is reduced tremendously. Additionally, any joint optimisation strategy would not lead to better results given the specific structure of the depth images. Besides extracting the data for SBHD, we will also discuss in the following how to store it efficiently.

Segmentation

What we want to achieve is to split the image into disjoint regions such that homogeneous diffusion inpainting is able to recover its content as good as possible. This means that the image within each of these regions should be smooth. At the same time, the sharp edges in a depth map ideally coincide with the segment boundaries.

Due to the simplicity of the depth images, the following straightforward two-step approach is sufficient to obtain the desired segmentation. In contrast to potentially more sophisticated approaches, this method is much easier to implement and the algorithm requires practically no runtime. First, a region growing algorithm is applied to get four-connected segments. A threshold T_1 thereby determines whether or not a neighbouring pixel belongs to the same segment. Small values of T_1 usually lead to a slight over-segmentation. Therefore, in a second step, we consider a Gaussian smoothed version of the original image with a standard deviation of σ. In this image we compute the mean contrast between adjacent pixels along a contour separating two neighbouring segments. Successively we remove the boundary with the lowest average contrast. We repeat this as long as the lowest contrast is smaller than some

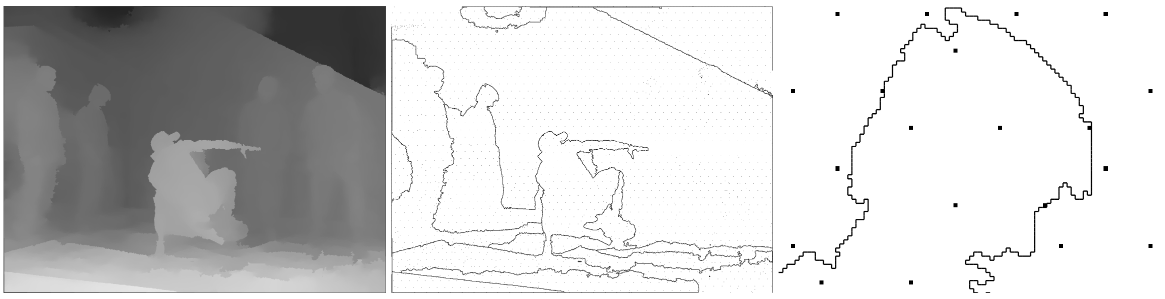

Figure 6.1. Contour extraction example. **Left:** Original depth image *breakdancers* (size: 1024 × 768). **Centre:** Extracted closed contours between pixels gained by the segmentation ($T_1 = 1$, $T_2 = 5$, $\sigma = 0.5$). **Right:** Zoom at the contour of one head with better visibility of mask points.

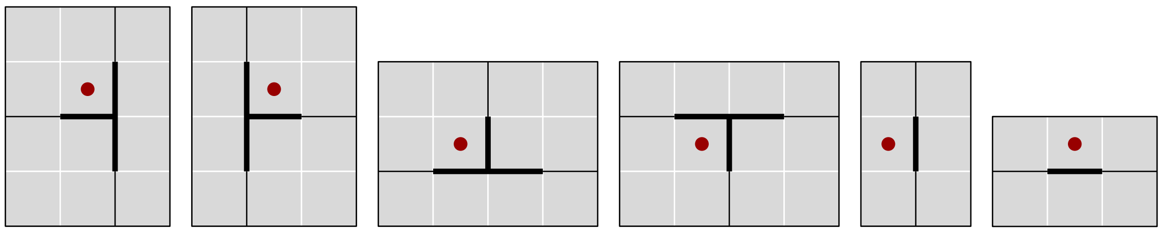

Figure 6.2. Different types of edge crossings along with their respective reference point (red dot).

threshold T_2. This method yields very precise contours between adjacent segments while allowing smooth transitions not to be split.

As already mentioned the segmentation yields closed contours at between-pixel locations. An example depth image along with the extracted contours is depicted in Figure 6.1. As desired, there is no contour around the two people on the right hand side of the image because of the smooth transition to the background. It is the task of data points within the segments to reconstruct such smooth transitions.

If one wants to use an existing codec like the JBIG (*Joint Bi-level Image Experts Group*) standard [102] to encode the contour information we would have to store two binary images, i.e., the edges in x- and y-direction, respectively. This however would be a huge overhead. Alternatively, we store this information more efficiently with a chain code. The advantage of between-pixel edges is that there are only three possible directions, whereas we would have seven directions when considering pixel chains. Thus, the chain code is highly efficient.

In the beginning we extract all T-junctions in the edge map, see Figure 6.2, types *i*-*iv*. To store them, we have to save the respective reference point coordinates along with the type number. Starting at these crossings we can build a chain code and stop whenever we reach an edge that has already been visited. Not needed starting elements can be removed afterwards. The only thing which remains are contours without any crossing. Therefore, we add two more starting element types representing only one edge, either in x- or in y-direction, respectively (see Fig. 6.2, types *v* and *vi*). The corresponding chain code has to be stored as well. Afterwards we employ a sophisticated lossless context-based entropy coder to encode the obtained edge information, namely the PAQ compression [123], version PAQ8o6.

Mask Points

Besides the segment boundaries in the image, we also have to encode the Dirichlet data for being able to reconstruct the image within the individual segments. As discussed in Chapter 5, it makes sense to first choose the locations of the mask points before selecting their tonal values for the selected optimisation strategy: An early optimisation in the tonal direction can hinder the spatial optimisation. Thus, we first concentrate on the mask locations. Here, we use a mixture of regularly sampled mask points and points at freely chosen positions as described in the following. This allows to obtain a trade-off between data optimality and coding costs. The overall amount of selected mask pixels is a free parameter which has a major influence on the final compression rate.

Initially, we choose points according to a specific fixed pattern as basis for the mask. The important property of this pattern mask is that it has a negligible coding cost. For instance, one possibility would be to uniformly sample random points over the whole domain given a specific seed. A drawback of this method is that there are specific areas where points are clustered, i.e. the distance between neighbouring points is not equal. Another approach would be to use mask points at a regular grid as done in [81]. However, this is still not optimal. Ideally, we seek a mask where the minimum distance between two distinct points is maximised. This allows to have a good covering of the whole image domain. Assuming no image boundaries are present, it is known that the hexagonal packing is the optimal one in the two-dimensional Euclidean plane, see [47]. Thus, we make use of such an hexagonal grid pattern. In order to make sure that there is at least one mask point in each segment, we compute the hexagonal mask separately for each individual segment. Thereby a certain density value determines the number of points.

In addition to the hexagonal mask we store some more free mask points. The larger coding costs of those points is justified by an expected large quality gain. From the presented methods in Chapter 5, we employ the probabilistic densification in combination with the nonlocal pixel exchange for selecting the additional points. These methods have several advantages compared to the other methods. In contrast to the method based on the theory of Belhachmi *et al.*, the densification can handle an existing mask in a straightforward way. Moreover, these two approaches allow to directly specify the desired mask density, in contrast to the optimal control approach for example. Methods relying on the discrete Green's functions are not as easy to apply here as they would have to be adapted to the arbitrarily shaped segment contours.

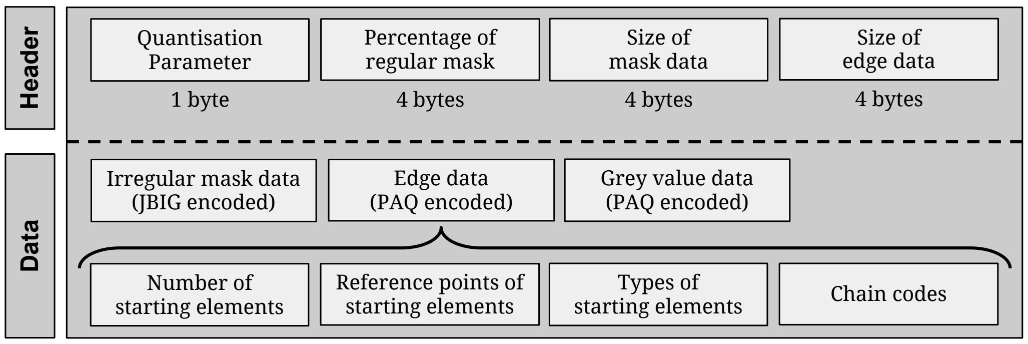
Header
Quantisation Parameter
1 byte
Percentage of regular mask
4 bytes
Size of mask data
4 bytes
Size of edge data
4 bytes
Data
Irregular mask data (JBIG encoded)
Edge data (PAQ encoded)
Grey value data (PAQ encoded)
Number of starting elements
Reference points of starting elements
Types of starting elements
Chain codes

Figure 6.3. File structure of the proposed codec.

The nonlocal pixel exchange already yields very satisfactory results already after only 1000 iterations. Since only a very small amount of mask points are required in typical compression scenarios (between 0% and 2%), one obtains a very fast method. The pixel exchange part can even be left out if one is interested in a faster encoding method with lower quality. While only having to store one density parameter for the hexagonal mask, we encode the location of the additional free mask points with the JBIG encoder [102].

Tonal Values

After the determination of the mask locations it clearly makes sense to optimise the tonal values at those positions. To allow for an efficient encoding, a quantisation to q different values is incorporated. To this end, we rely on the method presented in Section 4.9. As we have seen, it is possible to efficiently obtain the tonal optimised values for all different values of the quantisation parameter by precomputing the inpainting echoes once in the beginning. Note that, as we are using SBHD, the computations can be sped up by computing the optimal grey values for the individual segments in parallel.

The resulting optimised and quantised values at mask locations are then sorted in a list. The values for mask points belonging to the same segment are supposed to be next to each other. This has the advantage that subsequent values in are more likely to be similar due to the design of our segmentation, leading to a more efficient encoding. Finally, we again use the PAQ encoder to obtain a very compact representation.

Overview of File Structure

We are now able to write all the gathered information into one file having the structure as depicted in Figure 6.3. Note that we do not have to store the image dimensions in the header since they are already contained in the JBIG mask data.

6.1.3 Decoding

In the decoding phase we can follow a straightforward process chain. First of all we scan the stored header information and split the main data into the edge data, the additional mask, and the tonal information. Segment contours can be restored by placing the starting points and following the individual contour chains. Afterwards, we compute the hexagonal mask and add the irregular mask points to it. The grey values are placed at the corresponding locations, and we finally obtain the reconstruction via SBHD.

6.1.4 Parameter Selection

The presented SBHD compression algorithm incorporates many degrees of freedom given by the individual parameters. There are the thresholds T_1, T_2 and smoothing parameter σ for the segmentation, the percentage of overall mask points, the corresponding fraction of hexagonal mask points as well as the quantisation parameter for the tonal values. All of these parameters are ideally chosen such that the resulting data can be stored efficiently while at the same time giving qualitatively good reconstructions. Unfortunately, the influence of the chosen data on the final performance of the entropy coders is not obvious due to their non-transparent behaviour. Moreover, the mutual influence of the ingredients even requires to find a good balance between the individual parameters. For instance, assume we have two different configurations of the data which both yield the same reconstruction quality in terms of some error measure: On the one hand, we have a denser mask along with a very coarse quantisation, and on the other hand a sparser mask with more precise tonal values. It is hard to tell which one can be stored more efficiently given the unpredictable encoder performance.

We can remedy this issue of finding suitable parameters with the following strategy. First, we identify parameters which can be fixed a priori. This helps to reduce the amount of freedom. For example, we can compute the segmentation first without considering the following chain by manually checking if the contours seem to be reasonable. For the remaining parameters, the best we can do is to actually perform the complete compression with many different parameter combinations. For each combination we obtain a certain compression rate which is typically measured in *bits per pixel (bpp)*, as well as the corresponding PSNR error value which tells us how good the reconstructed depth map is. The parameters which yield the best reconstruction quality for a desired compression rate is then selected. This strategy also allows us to have a closer look at the individual parameters. By fixing a certain parameter, the remaining parameters are then chosen to obtain the best reconstruction quality for a certain compression rate. This is done in the following for the fraction of hexagonal mask points and the quantisation parameter. Thereby, we consider the depth map called *breakdancers* from the MVD sequences in [224] depicted in Figure 6.1.

An overview of using different percentages of the hexagonal grid points is shown

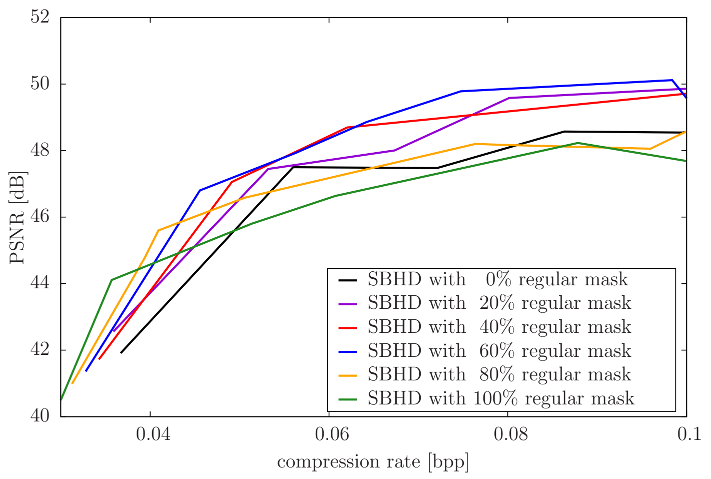

Figure 6.4. Compression with SBHD: Comparison of different percentage of hexagonal mask.

in Figure 6.4. As one can observe, higher percentages of the fixed mask should be chosen for smaller compression rates. This is clear, because the very precise location is not important when only a few mask points are present. In this scenario, no local details can be recovered anyways, so a slightly suboptimal mask is favoured. This allows to have extremely low coding costs for the mask, such that very small compression rates are feasible. On the other hand, smaller percentages are better for higher compression rates. This also makes sense: At some point, the hexagonal grid has sufficiently captured the image domain. In order to recover fine details, it is necessary to have specifically chosen mask points for those features. Between these two extremes, a continuous transition explains the plot. In total, this shows that the percentage of the hexagonal mask points cannot be determined a priori, but depends on the actual compression rate.

The results concerning the quantisation parameter are depicted in Figure 6.5. Here, it is obvious that smaller quantisation values do not yield a good trade-off between coding cost and quality gain. Especially for $q = 20$, the amount of different tonal values are apparently not sufficient to represent the smooth transitions in the image. Again, it seems that the best quantisation value again depends on the compression rate. If one wants to fix it a priori, a value between 60 and 80 seems to be a good choice.

6.1.5 Extension to Biharmonic Operator

A straightforward modification of SBHD is to use the biharmonic operator instead of the Laplacian. This means that the Equation (6.1) is replaced by

$$\Delta^2 u = 0. \tag{6.2}$$

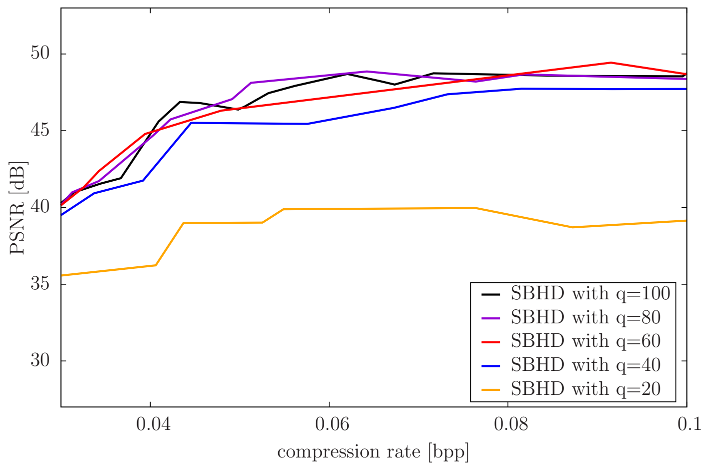

Figure 6.5. Compression with SBHD: Comparison of different quantisation values.

As a result, one obtains the so-called *segment-based biharmonic diffusion (SBBD)*. From Chapter 5, we know that the biharmonic operator has a much higher reconstruction capability compared to the harmonic counterpart, given the same amount of mask points. Thus, it makes sense to evaluate the performance of this operator in a compression framework as well. This modification can be incorporated in the whole compression algorithm in a straightforward way: Every time SBHD would be used, the SBBD is used instead.

Given the best possible parameters for each compression rate, Figure 6.6 depicts the resulting performance of both variants. Interestingly, this comparison clearly shows that the harmonic operator is the favoured one compared to the biharmonic operator when it comes to the compression performance. Even though the biharmonic operator has a higher reconstruction capability, it suffers much more from the suboptimal Dirichlet data which arise from the hexagonal mask and the quantisation of the tonal data. This will also be confirmed by the more extensive analysis in Section 6.2. Thus, we will not consider the biharmonic operator any further in the present compression algorithm.

6.1.6 Comparison to Standard Approaches

Now, we want to see how good the SBHD method performs compared to standard methods such as the well-established standard JPEG or its successor JPEG2000. Further, we will also conduct a comparison to the designated future standard HEVC (High Efficiency Video Coding), version HM-8.2. Although this codec is designed for the purpose of video coding, it also provides an intra-coding mode for the efficient compression of still images [145]. Besides the image *breakdancers* we further consider the test image *ballet* from the MVD sequences in [224].

Figure 6.7 depicts the qualitative results for a compression rate of 0.045 bits per

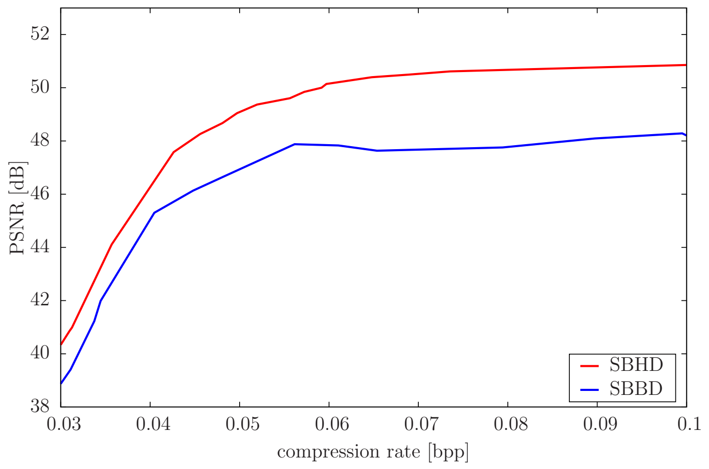

Figure 6.6. Compression with SBHD and SBBD: Comparison of harmonic and biharmonic operator.

pixel (bpp), which roughly corresponds to a compression ratio of 180 : 1. The overall mask density in this case is 0.3% and 0.45% for the images *breakdancers* and *ballet*, respectively.

The transform-based methods JPEG and JPEG2000 often perform well when it comes to the compression of standard natural images. However, both methods suffer from artefacts around edges. When it comes to depth images, these block or ringing artefacts around object boundaries are visually perceived much more unpleasant than in smooth image regions. HEVC seems to overcome these problems, but tends to smooth out some of the edges, which can lead also to a distorted geometric perception. With our SBHD method it is hard to notice any difference between the original image and the reconstruction.

A quantitative comparison is shown in in Figure 6.8. Note that for JPEG it is not possible to reach very high compression rates. One can see that our method clearly outperforms JPEG and JPEG2000. Our codec is even able to exceed the quality of HEVC for some compression rates.

Furthermore, we also evaluate the results considering a more perceptual error measure like the structural similarity index (SSIM) [203]. We use the available MatLab version with standard parameters from [203]. Figure 6.9 depicts the results. Note that a SSIM closer to 1 denotes a better visual similarity. Compared to JPEG or JPEG2000, it is visible that our proposed method reaches better results for almost all tested compression rates. HEVC and SBHD give reconstructions of comparable quality. This result is remarkable since, in contrast to HEVC, our algorithm makes use of relatively simple and straightforward concepts. We thus believe that it has potential for further improvement.

In our current proof-of-concept implementation, it takes several minutes for an image to be encoded, depending on its size and the number of mask points. The

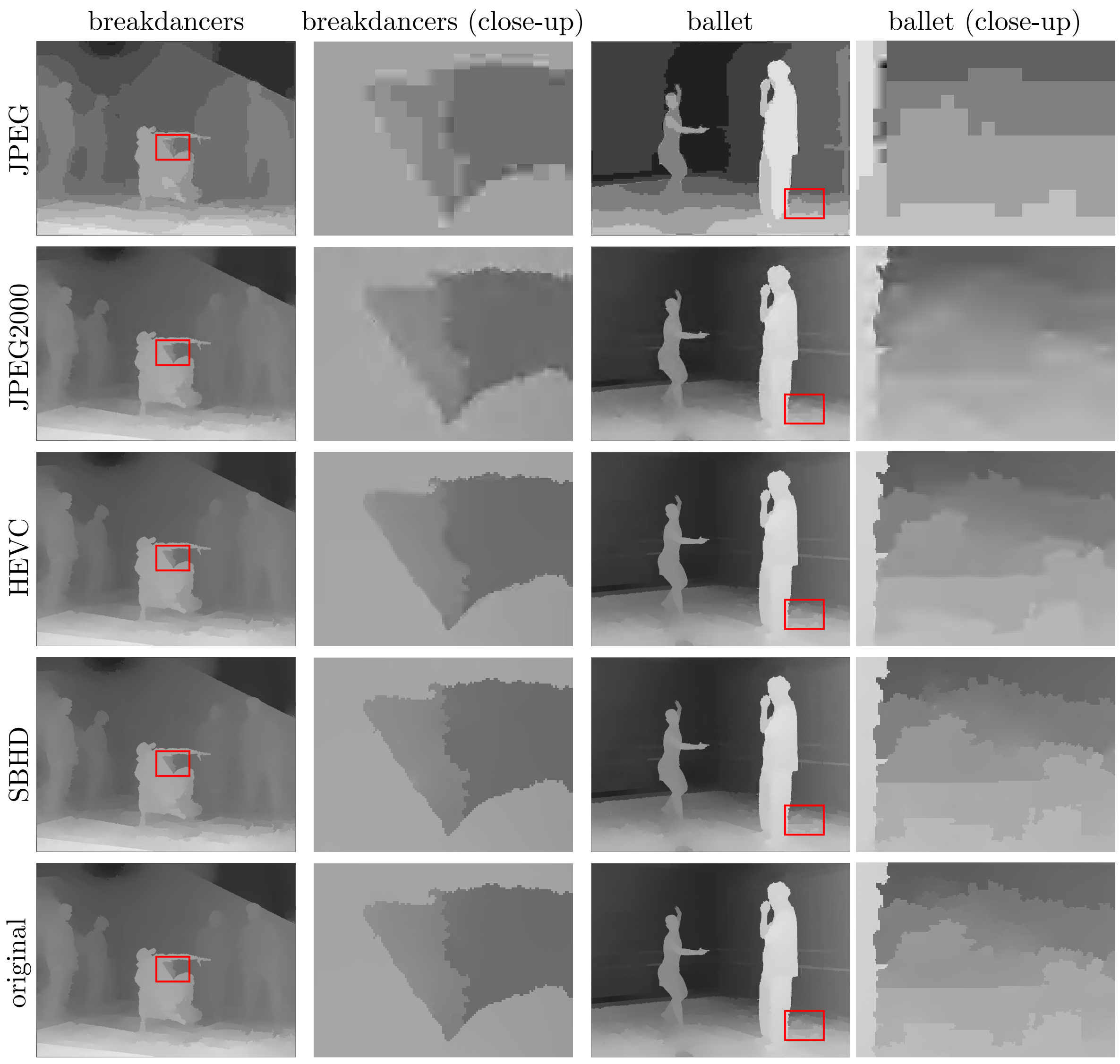

Figure 6.7. Comparison of different compression methods for two depth images using a compression rate of 0.045 bpp. The boxes denote the area of the respective close-ups.

decoding of a depth image of size 1024×768 can be done within a second on a modern PC. It is important to mention that there is a lot of potential for accelerating this process. For example, one can incorporate the bidirectional multigrid strategies into SBHD [125]. In this way, we expect that real-time decoding becomes feasible for practical applications, depending on the runtime of the entropy coders.

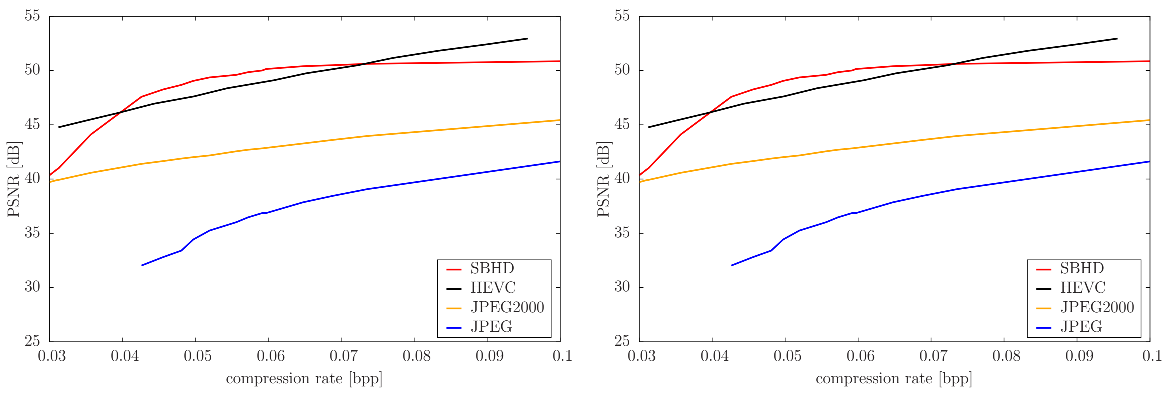

Figure 6.8. Quantitative comparison to JPEG, JPEG2000 and HEVC. **Left:** Image *breakdancers*. **Right:** Image *ballet*.

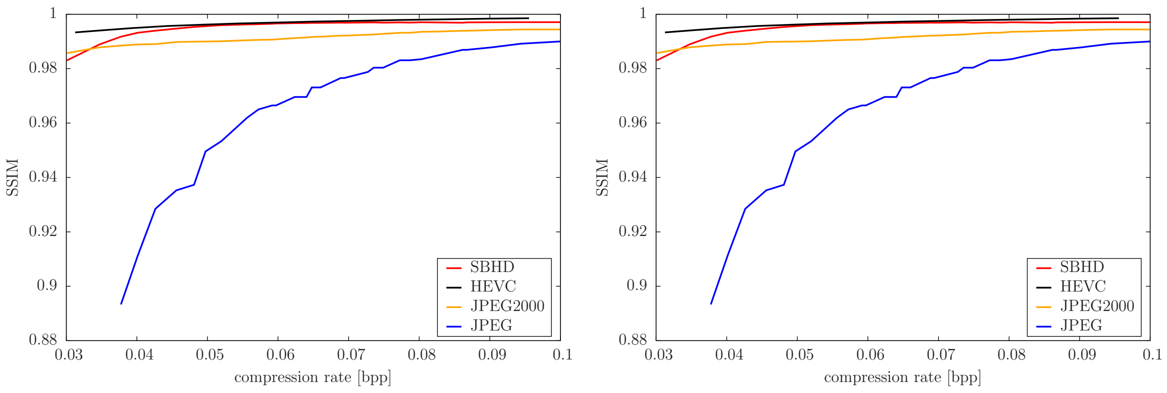

Figure 6.9. Perceptual comparison to JPEG, JPEG2000 and HEVC using the SSIM measure. **Left:** Image *breakdancers*. **Right:** Image *ballet*.

6.1.7 Conclusion and Outlook

We have shown that a combination of two relatively elementary concepts can lead to a remarkable compression quality of depth maps: a region-growing segmentation method as well as a homogeneous diffusion inpainting with carefully selected data points. Thereby, an interesting result is that the biharmonic operator does not lead to a better compression performance than the harmonic operator, even though it gives favoured reconstructions when no quantisation is used. In our evaluation, the segment-based homogeneous diffusion (SBHD) codec clearly outperforms JPEG and JPEG2000. Moreover, it performs competitively with HEVC, especially in terms of perceptual quality.

In the future, one can think about more sophisticated ways to find the optimal data to be encoded. For instance, it is possible to use an appropriate energy functional to obtain a good trade-off between segment boundaries and mask points. To this end, a combination of the well-known Potts model and the optimal control approach seems promising. While the presented separate optimisation yielded very good results for the test images, such a joint model could especially be helpful for noisy depth images. As discussed in Section 6.1.4, it would be good to know the actual coding costs of the individual ingredients given some entropy coder. This could be beneficial in choosing the right balance between the amount of contours, mask points and quantisation levels. Finally, instead of solely compression the depth images, the compression algorithm could benefit from additional information such as multiple views, combined colour / depth images, and their temporal extensions.

6.2 High Quality Compression of General Images

After focusing on a dedicated compression algorithm for depth images, we also want to consider arbitrary images. As mentioned in Section 1.2.3, there already exist successful PDE-based image compression algorithms for that purpose. In particular, the R-EED algorithm by Schmaltz *et al.* [170] marks the current state-of-the-art and is capable of outperforming JPEG and even JPEG2000. However, all successful PDE-based compression methods in that field rely on EED inpainting. Linear inpainting operators are mainly used for dedicated image types such as cartoon-like images or depth maps so far and did not achieve a breakthrough for general image compression yet.

In Chapter 5, we have seen that there exist various techniques to optimise data for image inpainting. Their main objective is to find a mask of a certain density along with continuous tonal values, and do not take into account any kind of coding costs. Thus, despite the promising performance of those optimisation approaches, it is not clear if their results can be exploited for an actual competitive compression framework.

In the following, we want to evaluate the potential of linear inpainting operators as well as EED inpainting in a compression context. To this end, we combine the sophisticated data optimisation strategies with efficient entropy coding methods. For linear inpainting, we rely on the best masks given by the optimal control approach presented in Section 5.2. When it comes to EED inpainting, the probabilistic sparsification approach along with the nonlocal pixel exchange is employed to optimise the spatial data, see Section 5.7. While an initial version of this new compression framework has been presented at the Pacific Rim Symposium in 2016 [5], an extended version including EED inpainting was published in 2016 [6].

In the following, the compression algorithm is presented. First, the encoding and decoding procedure is explained in Sections 6.2.1 and 6.2.2. Afterwards in Section 6.2.3, we will evaluate the proposed compression method, and also compare it to other approaches. In the end, a conclusion and outlook is given in Section 6.2.4.

6.2.1 Data Extraction and Encoding

Based on the aforementioned idea, the new compression algorithm is presented in the following.

Mask Points

In a first step, we obtain an optimised mask that contains a certain percentage of mask points. For linear inpainting, we use the optimal control approach from Section 5.2, and for EED inpainting the probabilistic sparsification approach followed by the nonlocal pixel exchange, see Section 5.7. The mask density is used to steer the compression rate. In the end, a binary image needs to be encoded. In [6], several entropy coders have been tested for that purpose. It could be experimentally shown that PAQ [123] in combination with a block coding scheme [220] leads to a more efficient coding compared to other methods like JBIG [102], JBIG2 or DjVu [28]. This block coding proposed by Zeng and Ahmed is specifically designed for sparse binary patterns. It decomposes the signal into blocks of a certain length, and encodes the relative position of each occurring 1-bit.

Tonal Values

Secondly, we have to find optimised tonal values at the fixed mask points. As in the depth map compression, we incorporate a quantisation to obtain an efficient compression scheme. Again, a quantisation parameter q specifies the number of different grey-values. For the linear inpainting operators, the quantisation-aware tonal optimisation is performed with the presented algorithm in Section 4.9. This again allows to efficiently test various values of the quantisation parameter with the help of the inpainting echoes. Unfortunately, this algorithm is not applicable for the tonal optimisation with EED inpainting. As a remedy, a simplistic approach from the R-EED method is used. It visits all mask points in a random order and checks if increasing or decreasing the tonal value to the next quantised values yields an improvement. If so, the new pixel value is kept, otherwise the value is reverted to the original one. This procedure is iterated several times until convergence. Even though this method is rather slow, it is in principle applicable to any inpainting operator.

To ease the issue of selecting the appropriate quantisation value, the following strategy is followed. The entropy coding of the grey values becomes more efficient for smaller numbers of different grey values. Thus, a smaller q leads to a smaller file size in general. At the same time, the error increases due to less accurate tonal values. Hence, 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 $\text{MSE} : \{0, ..., 255\} \to \mathbb{R}$ the corresponding error. By normalising both quantities to the range $[0, 1]$ and combining them additively, we define the *trade-off coefficient* μ:

$$\mu := \frac{S(q)}{S(255)} + \frac{\mathrm{MSE}(q)}{\mathrm{MSE}(255)}. \tag{6.3}$$

The smaller this coefficient, the better the trade-off for a given q. Thus, for each mask, we pick the q which yields the smallest μ.

Overview of File Structure

In total, the final compressed image file is created as follows. Besides the header containing the image dimensions and the quantisation parameter, the block coded mask as well as the tonal data information are written into a file. Finally, the complete data is encoded using the PAQ coder.

6.2.2 Decoding

The decoding is very straightforward. After applying the PAQ decoder, the header information is parsed. Then, the mask information is decoded and the tonal data is assigned at the corresponding mask points. Finally, an inpainting is performed with the appropriate inpainting operator.

6.2.3 Experiments

We are now in a position to evaluate the performance of the three different inpainting strategies in a compression framework, namely the harmonic, biharmonic and EED inpainting. First of all, we will investigate which of the individual inpainting operators is best suited in a compression context. Additionally, we have a closer look at the optimised inpainting data and investigate how quantisation artefacts influence the reconstruction quality. Afterwards, the compression algorithm is compared to other methods such as the JPEG, JPEG2000 and the R-EED algorithm.

Compression Results

In the beginning, we have a look at the best possible result which can be obtained by the optimisation approaches. To this end, we find masks of varying density for the individual inpainting operators, and perform a tonal optimisation afterwards. Thereby, no quantisation is used initially. All the initial experiments are performed on the test image *peppers* of size 256×256. The left graph in Figure 6.10 shows the corresponding error values of the reconstructed images by the individual inpainting methods. As already found out in Chapter 5, the biharmonic operator has a higher reconstruction capability than the harmonic one, while EED inpainting surpasses both linear counterparts. This observation is consistent throughout all mask densities. Note that when the mask becomes more dense, the differences between the operators

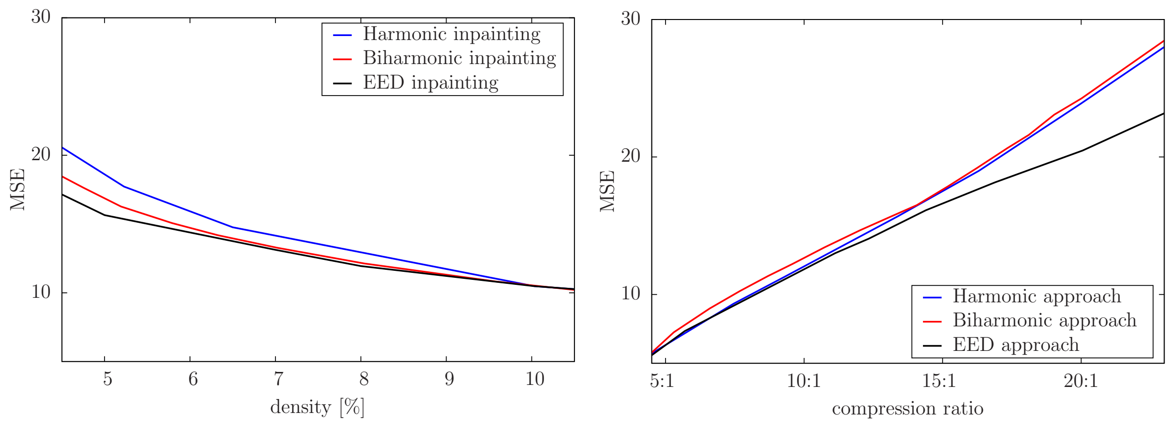

Figure 6.10. Experimental results for image *peppers*. **Left:** Error values inpainting after optimising spatial and tonal optimisation for different inpainting operators and mask densities. **Right:** Resulting compression performance based on optimised data.

become smaller. To obtain a visual impression, Figure 6.11 depicts the results of the masks with a density of 5%.

In a next step, we consider the actual compression performance with the individual inpainting operators. The resulting graph is shown on the right in Figure 6.10. It is visible that compression with EED inpainting (EED approach) cannot be outperformed by a compression routine relying on a linear inpainting operator (harmonic or biharmonic approach). In particular for higher compression ratios, EED inpainting benefits from the advantageous edge-enhancing behaviour which enables to reconstruct fine structures even in the presence of only a few data points. Thus, it is the preferred choice in those situations. Nevertheless, the harmonic approach performs equally good as the EED approach for smaller compression ratios below approximately 15:1. In those scenarios, one can thus prefer the conceptually simpler harmonic inpainting which offers many advantages: For instance, its linearity allows a faster processing and enables to perform a well-defined tonal optimisation with the help of inpainting echoes. Moreover, it enables to apply the complete theory of discrete Green's functions from Chapter 3. Last but not least, the linear inpainting operators are completely parameter-free.

As another interesting observation, the harmonic approach has a much lower error compared to the biharmonic approach at the same compression ratio. Even though this seems counter-intuitive, there is a simple explanation for this observation. The quantisation leads to suboptimal tonal values at the mask points. Obviously, harmonic inpainting can handle this situation much better than its biharmonic counterpart. Apparently, biharmonic inpainting is much more sensitive to changes in the Dirichlet data. To support this explanation, let us consider the experiment in Figure 6.12. Here, the reconstructions with different quantisation values are computed for the different inpainting operators. As Dirichlet data, we use the optimised masks from

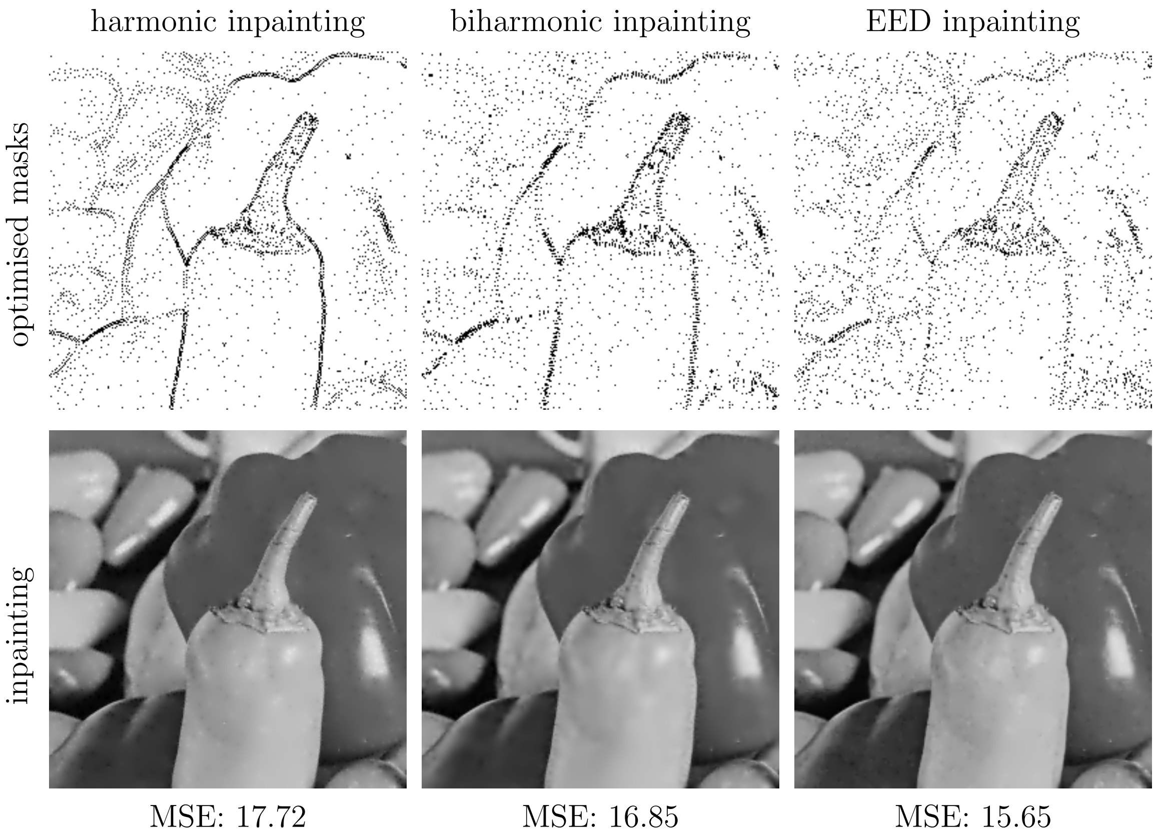

Figure 6.11. Example results of data optimisation strategies for image *peppers*. The optimised masks have a density of 5%.

Figure 6.11, and the corresponding grey-values are obtained by the quantisation-aware tonal optimisation. While the biharmonic inpainting outperforms the harmonic one when no quantisation is used, it shows a much more severe quality drop the coarser the quantisation becomes. On the other hand, the harmonic inpainting and the EED inpainting are clearly less sensitive against imperfect tonal data. Hence, we do not consider the biharmonic operator any further in the following.

Note that the same effect also occurred in the depth map compression (cf. Section 6.1.5). Here, not only the quantisation of the tonal values, but also the hexagonal mask locations lead to suboptimal Dirichlet data. This prevents the sensitive biharmonic operator from allowing any good compression performance.

Comparison to other Compression Algorithms

In the following , we compare the new algorithm to existing compression algorithms, namely JPEG, JPEG2000 and R-EED. Figure 6.13 shows the results for the image *peppers*. As one can see, the harmonic approach is able to outperform the standard methods JPEG and JPEG2000 and even the R-EED approach for low to medium

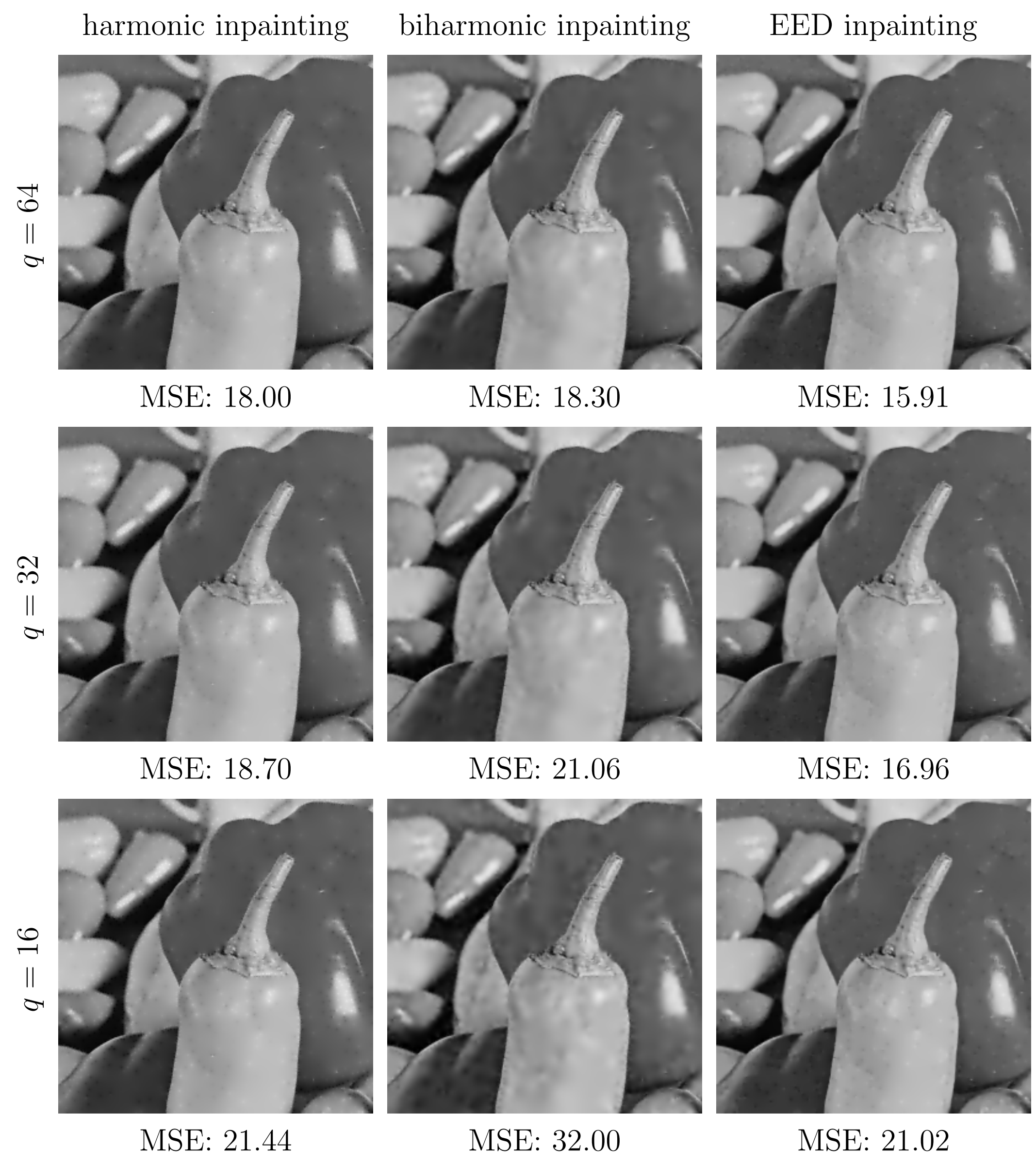

Figure 6.12. Influence of quantisation on inpainting result for different operators. The reconstructions are based on the masks shown in Figure 6.11 and the respective quantisation-aware tonal optimisation strategies.

compression ratios of up to 15:1. This is an astonishing result: It is possible to beat JPEG2000 and even R-EED for a general image with a compression method based on a linear PDE. The EED approach performs equally good in that range, and can even beat all other methods up to a compression ratio of almost 20:1. For higher compression ratios, the R-EED method takes over the lead.

Figure 6.14 gives a qualitative comparison of the methods for a compression ratio

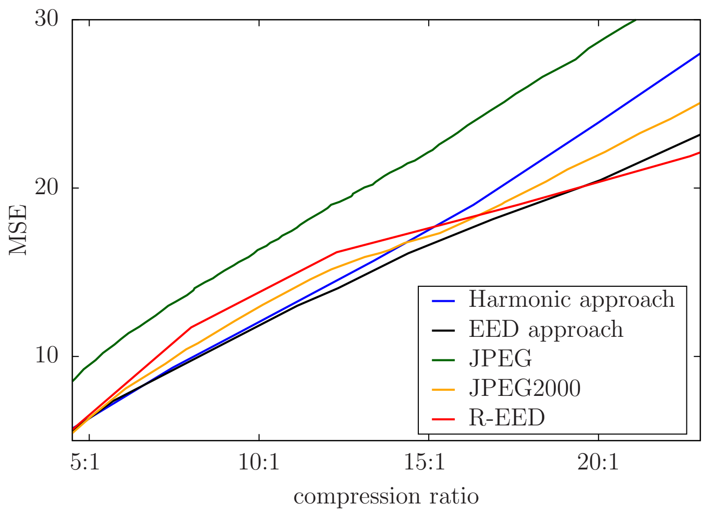

Figure 6.13. Quantitative comparison of different image compression approaches for the image *peppers*.

of 8:1. While JPEG and JPEG2000 show obvious artefacts at the edges, harmonic inpainting does not show such obvious effects. Compared to R-EED, it is much smoother in homogeneous regions. While EED inpainting tries to preserve some undesired edges, homogeneous diffusion inpainting gives smooth transitions by design.

The following Table 6.1 shows the potential of the harmonic approach for different images. For all images, the harmonic approach is able to beat JPEG, and is able to compete with R-EED and JPEG2000.

Table 6.1. Comparison of different compression approaches on several test images. A compression rate of 15:1 is used. The table shows the MSE of the respective reconstructions.

Image	elaine	lena	trui	walter
JPEG	34.69	20.06	16.24	6.73
JPEG2000	**31.23**	**14.73**	12.27	5.70
R-EED	35.48	16.56	11.27	5.48
Harmonic approach	31.38	17.00	**10.48**	**4.53**

A variant of the discussed approach is also presented in [6]. The main modification is that the mask is represented by a tree-subdivision structure like in R-EED. It allows a more efficient encoding at the cost of having less optimal mask locations. A newly proposed probabilistic tree densification and nonlocal node exchange are used to optimise the mask. In the end, the approach allows to outperform all other approaches for higher compression ratios of more than 30:1 with a biharmonic inpainting operator.

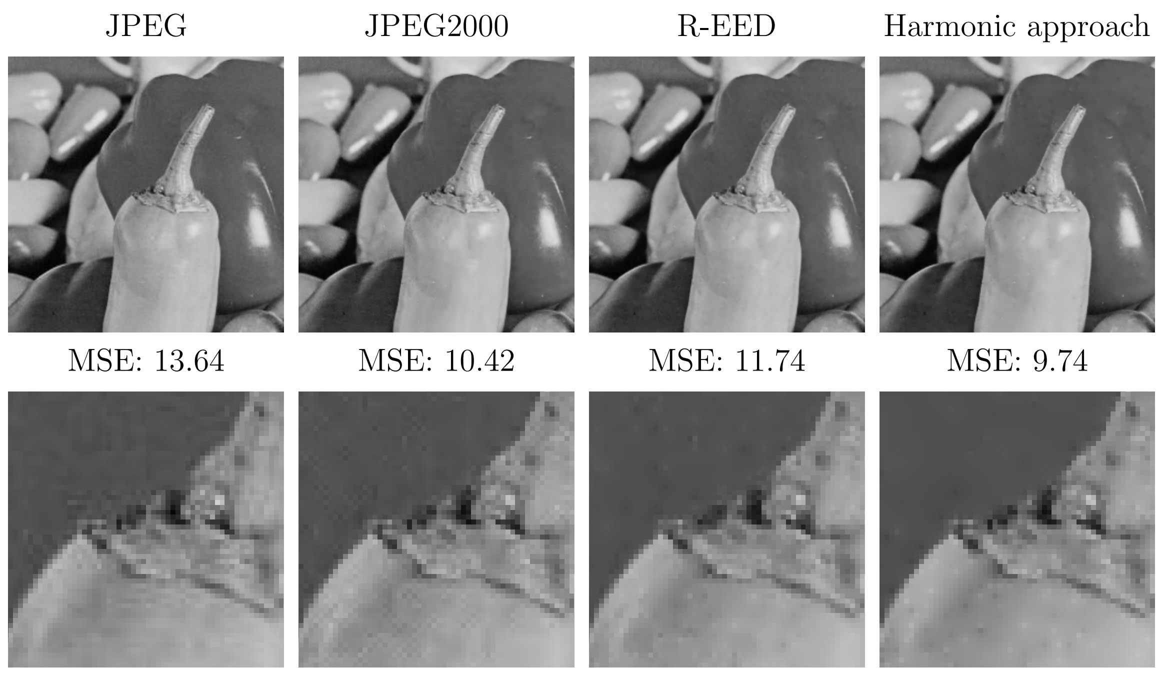

Figure 6.14. Qualitative comparison of different compression approaches for the image *peppers*. The compression ratio for all methods is 8:1. The images on the bottom show magnifications.

6.2.4 Conclusion and Outlook

We have seen that advanced data optimisation techniques play a major role for designing competitive compression algorithms. A remarkable result is that a conceptually simple linear PDE is able to yield the same compression performance as EED inpainting for medium to low compression ratios. Moreover, it can even outperform the standard method JPEG and JPEG2000 as well as the current state-of-the-art in PDE-based compression. For general images, this has never been achieved before. Considering all the advantages of linear inpainting operators, this opens whole new possibilities.

Despite these qualitatively very good results, the runtime of PDE-based compression algorithms typically remains an issue. Even though the inpainting step can be achieved in real-time with GPU support or even on the CPU for smaller images, the decoding speed of JPEG or JPEG2000 cannot be reached, yet. Thus, as a next step, we will try to become competitive to standard approaches in terms of both quality and speed in Section 6.3. Another aspect for future work is to extend the presented method to colour images.

6.3 Efficient Compression with Block Inpainting

In Sections 6.1 and 6.2 the power of linear PDEs in an image compression context was shown. We have seen that it is even possible to outperform standard methods such as JPEG, JPEG2000 or HEVC when it comes to the compression quality. The only major drawback of PDE-based compression algorithms is the runtime which is often used as main argument against them. In the following, we want to address this remaining issue by presenting a PDE-based compression algorithm which is able to compete with standard approaches in terms of both decoding runtime and quality.

To reach this goal, we adopt some basic strategies from JPEG. For instance, a central idea here is to decompose the image into smaller blocks. In combination with the powerful theory of discrete Green's functions, the hope is to obtain a huge speedup while at the same time yielding promising reconstructions. Moreover, we will analyse the influence of exchanging individual ingredients on the compression performance. For example, we consider different inpainting operators, different strategies to find the masks, different entropy coders, different representations of the tonal data and different quantisation strategies.

In the following, we will first introduce the concept of a block-based inpainting in Section 6.3.1. In this context, new optimisation strategies will be presented for this new inpainting strategy. This will allow to analyse its potential in more detail. Afterwards in Section 6.3.2, a first compression method based on the block inpainting is presented. In Section 6.3.3, an improved variant of this algorithm is proposed. Since we are interested in a fast decoding, acceleration strategies are shown in Section 6.3.4. An extension to colour images is done in Section 6.3.5. Finally, a conclusion and outlook is given in Section 6.3.6.

6.3.1 PDE-based Block Inpainting

First, we study the inpainting based on a block decomposition in more detail. After introducing this new inpainting technique, we will see that it allows perform various tasks such as inpainting or data optimisation in a very efficient way. Besides the improvement in terms of speed, we also consider the reconstruction capabilities as well.

Description

Let us start by introducing the new block inpainting. To this end, we assume the image domain to be decomposed into blocks of size $H \times H$. In our case, we set $H = 8$ for efficiency reasons, as we will see soon. If the image size does not equal a multiple of H, we extend the image to the next larger multiple of H for processing purposes by naturally mirroring the image. This allows to handle arbitrary image sizes. Note however that all comparisons are performed on the targeted original image domain by cropping the obtained image in the end.

For a given mask and corresponding tonal values at those locations, the block inpainting is performed in a straightforward way by performing an inpainting within the individual blocks, separately. Thereby, homogeneous Neumann boundary conditions are assumed at the block boundaries. Hence, the overall inpainting is strongly localised, and the influence of the Dirichlet data is restricted to the individual blocks. Further, we only consider the linear inpainting operators given by the powers of the Laplacian in the following. Note that this block inpainting can be seen as a special case of the segment-based inpainting presented in 6.1.1 where the segment boundaries are given on a regular grid.

Efficient and Accurate Inpainting

Obviously, the block decomposition has the advantage that the inpainting problem can be computed in parallel for all blocks. For each block, any aforementioned strategy can be employed for finding the inpainting solution, such as a bidirectional multigrid solver or a FED scheme. However, due to the limited amount of mask points per block, a better way is to rely on the approach using discrete Green's functions. This has two major advantages: First of all, the solution is found much faster due to the very small linear systems of equations that need to be solved. Secondly, the obtained solution is highly precise up to machine precision. In particular, it does not depend on some stopping criteria which in addition would have to be specified. Consequently, we find the overall inpainting solution by executing Algorithm 3.4 for all blocks. As a matter of fact, the algorithm can even be accelerated tremendously by exploiting the block decomposition as described in the following.

Due to the fixed block size of $H \times H$, we can use this knowledge as a prior. First of all, note that the Greens mother function (GMF) is the same for all blocks. Thus, it is possible to reuse it for all blocks once it is obtained. In the beginning, one can either compute it once, or directly hardcode it into the program. This allows a fast setup of the linear system of equations.

After solving it for the coefficients of the discrete Green's functions and the mean value constant, the inpainting is obtained with the help of the frequency transform as described in Algorithm 3.1. Since we assume homogeneous Neumann boundary conditions at the block boundaries, we have to use the discrete cosine transform (DCT). In particular, the choice of $H = 8$ allows to use the frequently used 8×8 DCT. For example, this specific transform represents the core part of JPEG. Due to its popularity, many work has been done in the past to optimise the 8×8 DCT to an extreme: By minimising the number of basic operations such as multiplications or additions, very time-efficient schemes have been proposed over the years. A very good overview is presented in [159]. In this work, we use the 8×8 DCT algorithm and its inverse of Vetterli and Ligtenberg [199] due to its favourable performance. Note that the 2D-DCT is separable, meaning that one can split it into two steps: First one computes the 1D-DCT for every row, and afterwards performs a 1D-DCT of the result along every column. Thus, a 2D extension is straightforward given a 1D-DCT.

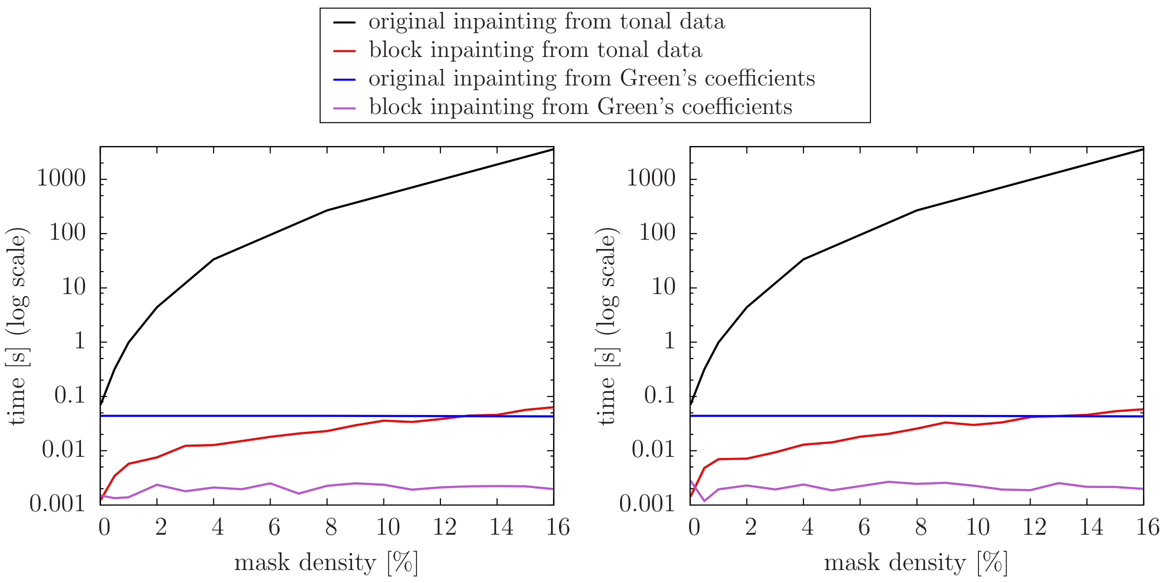

Figure 6.15. Runtime of block inpainting strategies compared to the original inpainting strategies for different mask densities. All strategies rely on discrete Green's functions, and use the tonal values of the image *trui* and masks given by the analytic approach of Belhachmi *et al.* as input. **Left:** Harmonic inpainting. **Right:** Biharmonic inpainting.

I want to thank Nirmal Kumar Ramadoss for the implementation support during his time as a student associate.

The beneficial performance of the new block inpainting is demonstrated in the following experiment. For the test image *trui*, we again consider different masks of varying densities obtained with the analytic approach of Belhachmi *et al.* These are the same masks as originally used for comparing different inpainting strategies in Section 3.5. Then we measure the runtime needed for performing a block inpainting and compare it to the required time for performing the original full inpainting with the same strategy relying on discrete Green's functions. Moreover, we also consider the sole runtime of Algorithm 3.1 to obtain the respective inpaintings from prescribed coefficients and constant. The results are shown in Figure 6.15.

Compared to the original inpainting strategies, the runtime improvement given by the block decomposition is remarkable. For smaller mask densities, one gains a speedup of approximately 100 for performing an inpainting from tonal values. For larger mask densities, one can even observe a speedup of 10000. The reconstruction from the coefficients is again obtained in constant time, but more or less 100 times faster than the original approach. It is clear that the runtimes do not depend on the order of the inpainting operator, and that the results are highly accurate up to machine precision. Given these very fast inpainting approaches, real-time decoding seems to be feasible.

Efficient Tonal Optimisation

Besides the inpainting, it is also possible to speed up the tonal optimisation by exploiting the block decomposition. Clearly, the individual blocks can be processed in parallel. For each block, we favour the approach relying on discrete Green's functions due to the small number of mask points per block, see Algorithm 4.5. In this algorithm, we again exploit the fact that $H = 8$ in the following way. First of all, we see that all possible entries of the matrix $\widetilde{\boldsymbol{H}}^\top \widetilde{\boldsymbol{H}}$ can be precomputed and hardcoded for the desired operators. This allows to quickly set up the system matrix. The right hand side of the normal equations equations is obtained with Algorithm 3.1 together with the fast 8×8 DCT. The same algorithm finally allows to obtain the tonal optimised inpainting.

As for the block inpainting, the following experiment demonstrates how fast the tonal optimisation can be computed. For the image trui and the same masks as before, the corresponding results are shown in Figure 6.16. Compared to the runtimes of the tonal optimisation strategies for the original inpainting, the tonal optimisation for the block inpainting is enormously faster. For example for a mask with a density of 4%, the new approach only takes around 0.01 seconds, while any other algorithm lasted more than 30 seconds. It is astonishing that the tonal optimisation is so fast that it could even be done in real-time, independent of the order of the inpainting operator. Since the tonal optimisation needs more or less the same runtime as an inpainting on a block decomposition, this offers completely new possibilities for the spatial optimisation as we will see in the following.

Efficient Spatial Optimisation

It is clear that all the performance gains with the block decomposition are only helpful if one can actually obtain a comparable reconstruction quality as with the original inpainting formulation. Hence, we try to find the best possible masks for block inpainting in the following. This way, we are able to judge the reconstruction capabilities of the block inpainting.

For finding the mask, we use modified versions of the probabilistic densification and nonlocal pixel exchange strategies. Besides their advantage of being able to specify the mask density explicitly, the block decomposition allows a very fast processing of the individual steps. In particular, this means that we are able to obtain a much higher optimised result due to an interleaved optimisation strategy.

Considering the probabilistic densification in Algorithm 5.2, the update of the inpainting is clearly the most time-consuming part. Fortunately, we can perform this step very efficiently in case of the block inpainting: After adding some points within a few blocks, we only need to update the inpainting in these blocks. The block inpainting remains the same for all other blocks. This is an enormous gain in speed. The same improvement can be applied to the nonlocal pixel exchange (NLPE) in Algorithm 5.3: After an exchange, the block inpainting has to be updated only in

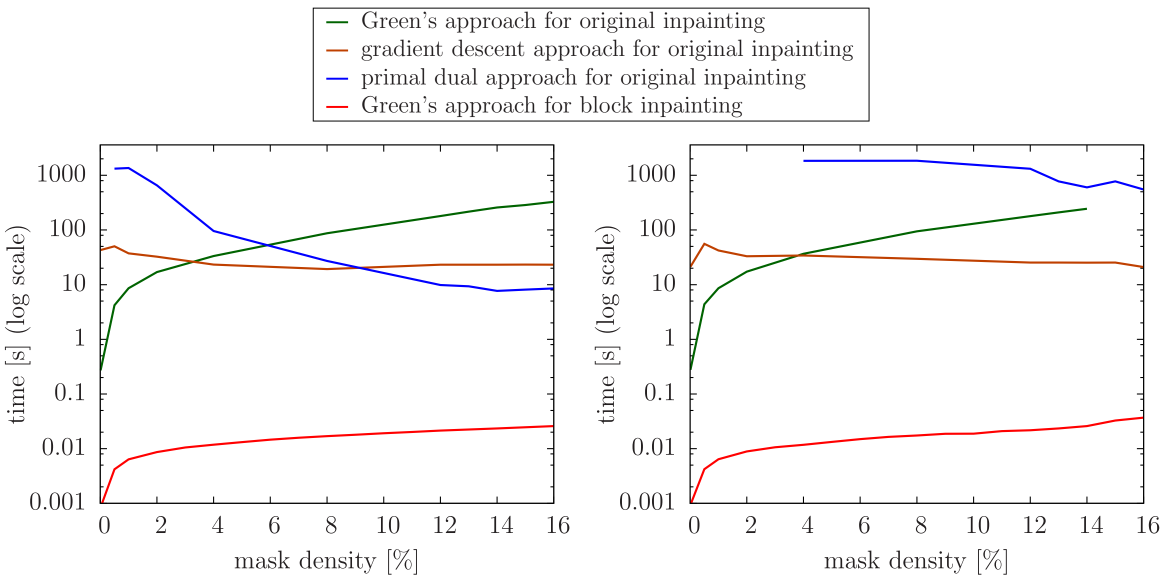

Figure 6.16. Runtime of tonal optimisation for block inpainting compared to the original tonal optimisation strategies for different mask densities. The image *trui* and masks given by the analytic approach of Belhachmi *et al.* are used as input. **Left:** Harmonic inpainting. **Right:** Biharmonic inpainting.

the blocks where the mask has changed. Moreover, the ability of performing a very fast tonal optimisation enables us to integrate it into the optimisation process of the densification or the NLPE, respectively. Thereby, instead of a block inpainting, a tonal optimisation is performed for all blocks where the mask has changed. We call these variants tonal optimised densification and tonal optimised NLPE. As a last variant, we perform all three parts interleaved, that is, a tonal optimised NLPE runs for several iterations after each densification step. Here, it suffices to perform 300 NLPE iterations after every densification step.

For the image *trui*, the results of the different variants are depicted in Figure 6.17. For all approaches, the PSNR value is shown after a tonal optimisation on the final masks for the sake of a fair comparison. First of all, we note that the ranking of the individual variants is identical for the harmonic and biharmonic operator. As already observed in Section 5.3, the tonal optimised densification does not yield good results: The optimisation process gets stuck in a bad local minimum due to the hopeless attempt of the tonal values to globally compensate for bad mask locations. Obviously, a subsequent tonal optimised NLPE helps to improve the result, but is still not able to get the mask completely out of the bad local minimum. Similar to before, performing a densification with a block inpainting instead of a tonal optimisation already gives much better results due to the smoothing effect. Interestingly, applying a NLPE after the densification does not necessarily lead to better results: The optimisation aims at the best mask given the original tonal values, and does not consider any tonal

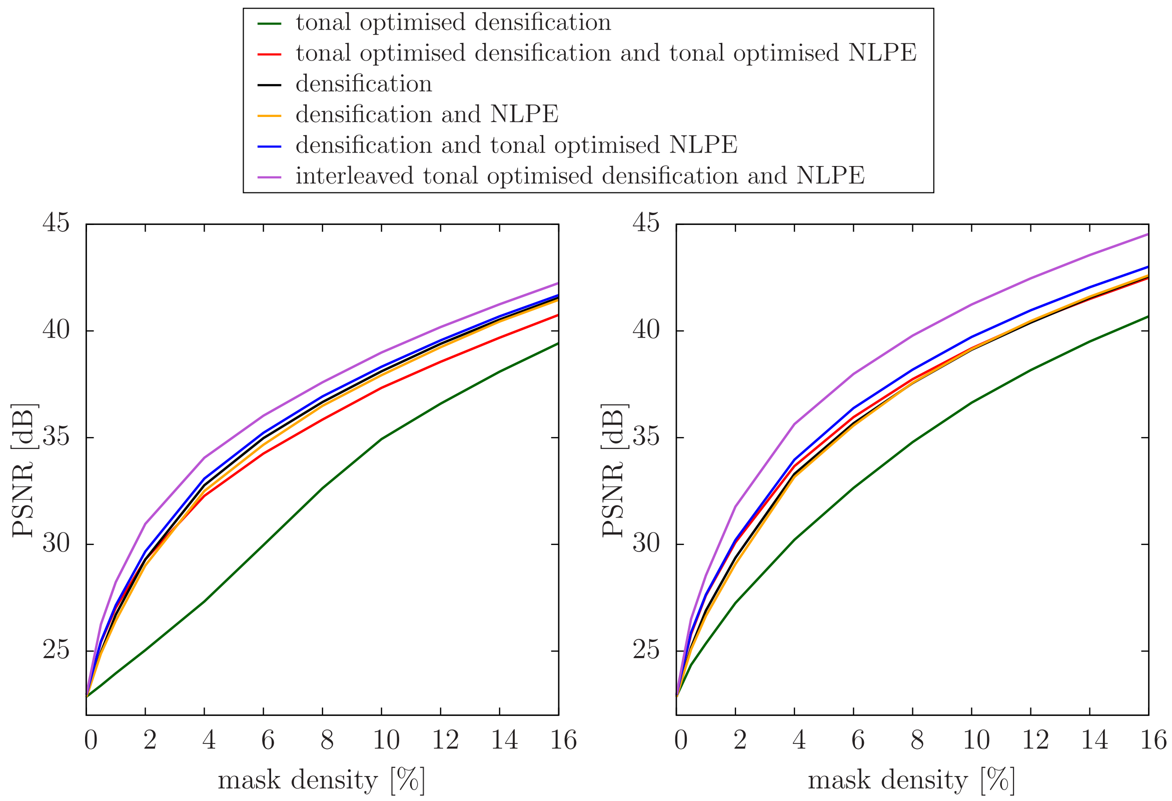

Figure 6.17. Spatial optimisation strategies for block inpainting. Masks of varying densities are found for the image *trui*. The PSNR is shown after performing a tonal optimisation on the respective final masks. **Left:** Harmonic inpainting. **Right:** Biharmonic inpainting.

optimisation. Thus, performing a tonal optimised NLPE after the densification helps to improve the mask. Finally, we see that the best result by far is obtained by a interleaved tonal optimised densification and NLPE. Obviously, it pays off to have a highly coupled optimisation process to avoid being stuck in a local minimum. Fortunately, the fast processing obtained by the block decomposition makes this approach feasible: Even for a density of 16%, the mask is obtained within some minutes. With the original inpainting formulation, this strategy would last several weeks.

Figure 6.18 shows an example of the best mask for the block inpainting obtained with an interleaved tonal optimised densification and NLPE. Compared to the results with the original inpainting from Chapter 5, the block based inpainting is able to offer a similar reconstruction capability. The optimal control approach is able to yield a better mask for a reconstruction with the original inpainting. Nevertheless, compared to the original densification and NLPE, the obtained mask for the block inpainting even leads to a higher quality reconstruction for the harmonic operator, and a comparable quality for the biharmonic operator. On a first glance, it is hard

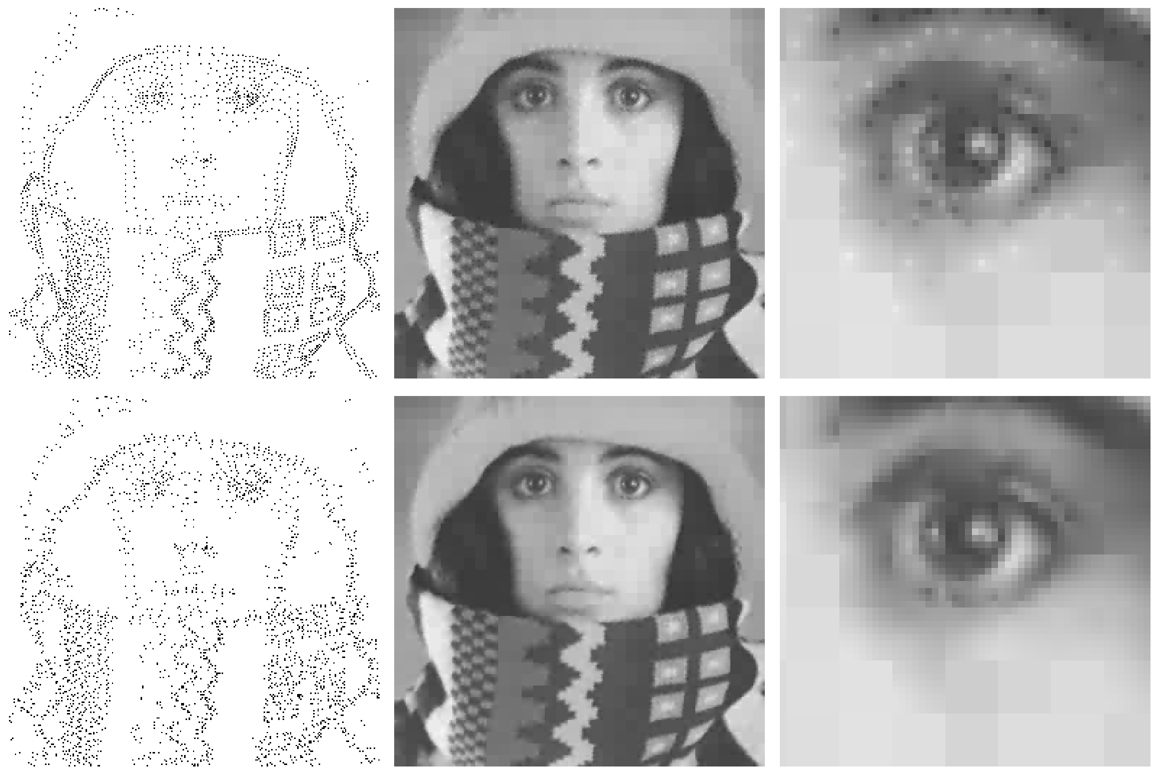

Figure 6.18. Spatial optimisation with interleaved tonal optimised densification and NLPE. **Top row:** Optimised mask (density: 4%) for harmonic block inpainting, corresponding tonal optimised block inpainting with zoom-in at left eye (MSE: 25.57). **Bottom row:** Same with biharmonic block inpainting (MSE: 17.77).

to recognise the block boundaries in the reconstructions. They only become clearly visible when considering the magnification.

Comparison of Inpainting Operators

Given the harmonic, biharmonic, triharmonic and quadharmonic operator, a natural question is which operator is best suited within a compression framework. To this end, we optimise the spatial and tonal data for all of these operators as good as possible for varying mask densities, and consider the resulting reconstruction qualities. Note that the theory of discrete Green's functions allows to switch between these operators by simply changing the power of the eigenvalues. Furthermore, we evaluate how a quantisation affects this quality for the individual operators. To this end, the optimal tonal data is found with the quantisation-aware tonal optimisation, see Algorithm 4.6.

The results are shown in Figure 6.19. If continuous values are used, i.e. without a quantisation, the harmonic operator consistently performs worst, and the biharmonic operator performs best. Interestingly, going from the biharmonic to the triharmonic

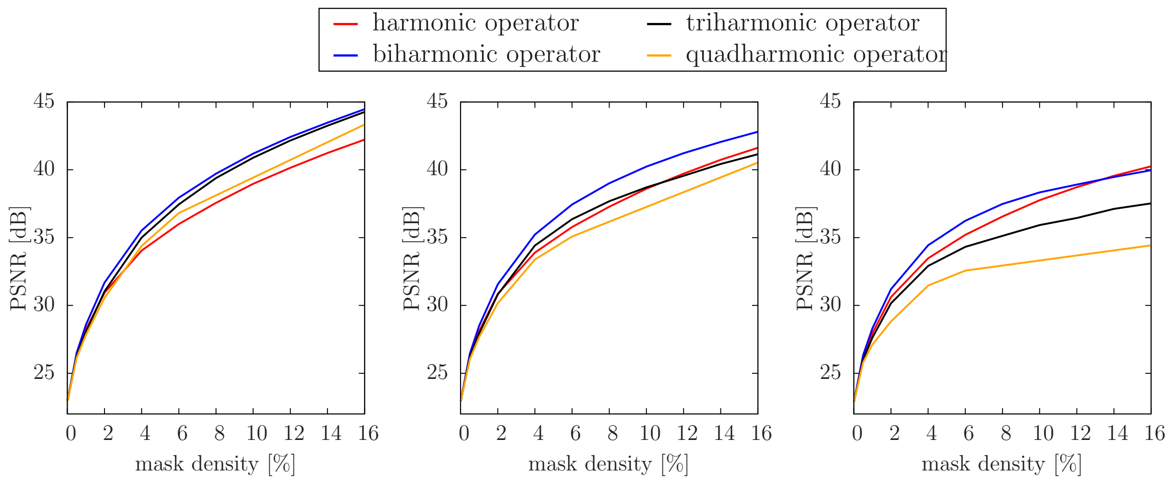

Figure 6.19. Comparison of best reconstructions obtained with different operators and quantisation levels. **Left:** No quantisation. **Centre:** Quantisation with $q = 64$. **Right:** Quantisation with $q = 32$.

operator does not improve the result. Instead, the quality of the reconstruction is comparable to the one of the biharmonic operator. The quadharmonic operator is even giving worse results than the triharmonic operator. Obviously, the higher order of the biharmonic operator compared to the harmonic one allows for a certain degree of flexibility. An even higher order seems not to be necessary for practical applications, and even leads to worse reconstructions due to the extreme over- and undershoots.

When quantisation is used, it seems that the quality deteriorates more for higher inpainting operator orders. This confirms the results in Section 6.2.3. In this experiment, we see that this observation also holds true for the tri- and quadharmonic operators. Those two operators are extremely sensitive to differences in the tonal data, especially for smaller values of the quantisation parameter. For this reason, we mainly focus on the harmonic and biharmonic operators in the following as they turn out to give the best reconstructions when quantisation is present.

6.3.2 Compression with Optimal Masks

The promising results of the block inpainting give rise to the hope to obtain an efficient PDE-based compression method in terms of decoding speed and quality. To this end, we use the highly optimised masks for a first compression algorithm. The structure of the compression algorithm is then straightforward: In the encoding phase, the header information including the image dimensions and a certain quantisation value is written into a file. Afterwards, the relative position of the mask points within all blocks are added to the file. An additional bit in the beginning of each block indicates if the block contains any mask point or not. Then, a quantisation-aware tonal optimisation is performed with the given quantisation parameter, and the tonal information is

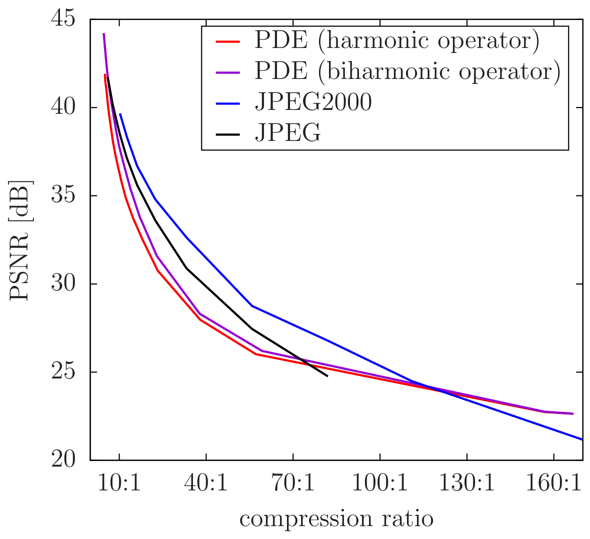

Figure 6.20. Compression performance results with block inpainting and optimal masks for different compression ratios for the image *trui*.

added to the file. Whenever there is no mask point within a block, the quantised mean value of the original image is stored. Finally, the PAQ entropy coder [123] is employed to reduce the file size. To find the best trade-off between quality and coding costs, the quantisation parameter is optimised as presented in Section 6.2.1.

In the decoding, PAQ retrieves the file content first. Then, the image with the mask points and corresponding tonal values is set up. The reconstructed image is then obtained by performing the block inpainting. Note that this compression algorithm is very similar to the approach presented in Section 6.2, with the major difference that the block inpainting is employed instead of the original inpainting formulation in combination with appropriate optimisation strategies.

The resulting compression performance for the image *trui* is shown in Figure 6.20. The varying mask densities lead to different compression ratios. We compare our PDE-based approaches with the harmonic and biharmonic inpainting operator to JPEG and JPEG2000. Unfortunately, we are not able to outperform these standard methods, except for their respective highest compression ratios. In contrast to the previous results in Section 6.2.1, the reconstructions given by the optimal control masks and the original inpainting are slightly better so that the compression performance of JPEG or JPEG2000 can be obtained. Thus, we have to find a more efficient way to perform the compression.

6.3.3 Compression with Subdivision Tree Masks

Since it does not pay off to store the highly optimised masks, we alternatively encode the masks with the help of an adaptive binary tree representation. This idea is very close to the strategy used in [6]. However, there are some important differences. First

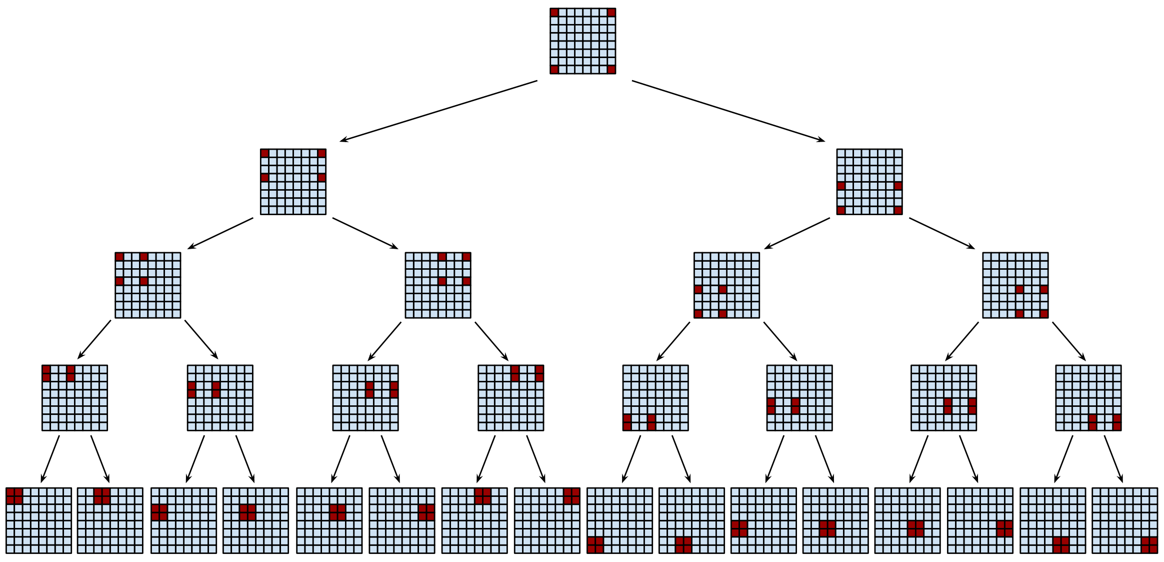

Figure 6.21. Binary tree structure for encoding the mask on a 8×8 block. The red points denote the corresponding mask points which are represent the corresponding node.

of all, we use individual trees for all blocks which enforces a certain structure in the mask points and thus leads to a more efficient encoding. Moreover, we will use an adaptive quantisation strategy which is also used in JPEG. These two modification will allow to obtain a more efficient compression algorithm.

Adaptive Binary Subdivision Tree Mask

A tree encodes a mask in the following way. All nodes of the tree represent a certain rectangular area of the block, and encode the 4 corner points as mask points. One starts with the root node representing the complete block. Whenever a node has two children, the domain is split into smaller parts. This splitting is repeated until only a 2×2 pixel area is left. If a node is not split further, it is denoted as leaf node. Given a tree representation, the overall mask is obtained by collecting the mask points from all leaf nodes. Thus, the mask contains more points in regions where the tree is split more often. This allows to insert more mask points in certain areas if it is required and hence to adapt to local image structures. Given the block size of 8×8, each tree thus has a relatively small number of possible configurations and hence 5 is the maximum number of levels. An overview of the tree structure for a 8×8 block is depicted in Figure 6.21. As an alternative, one can also make the rectangles slightly overlapping by one pixel when splitting a node. This avoids a clustering of the points along the split line. Since this was found to be more efficient [170], we will use this slight modification, too.

The tree representation of the mask can be encoded very efficiently as a bit sequence.

Starting at the root node on the top, each split decision is recursively encoded by a bit. Thus, if a node is split, a 1 is added to the sequence. Otherwise, if a node is a leaf node, a 0 is added. For some blocks, we also allow for blocks with no mask points. In this case an initial bit indicates if there is a mask and consequently a tree or not. Then, only the mean value of the original image will be encoded for this block.

For optimising the trees within the individual blocks, we again make use of the joint tonal optimised densification and NLPE. To this end, we adopt the stochastic tree densification and nonlocal node exchange strategies presented in [6]. Since we are dealing with a separate tree for every block, we make the following adaptation. In both algorithms, the nodes of all trees are considered together within one big set of nodes. If there is not yet a tree within a block, the whole block is considered as a leaf node and the corresponding error is computed for the whole block. If such a block is then selected for a splitting, a new tree consisting only of a root node is created. As before, this interleaved densification and node exchange in combination with an inherent tonal optimisation can be performed very fast with the block decomposition: We can rely on the discrete Green's functions to perform an efficient tonal optimisation within the blocks where the tree and consequently the mask has changed.

Adaptive Tree Quantisation

When one compares the compression algorithm to JPEG [150], one can observe that they have a lot in common. Not only do both approaches use a decomposition of the image domain into 8×8 blocks, they both make use of a frequency decomposition for each block. In JPEG, this frequency decomposition is obtained with the DCT, and the resulting DCT coefficients are used to encode the image. Interestingly, the binary tree subdivision mask of the PDE-based compression can also be interpreted as a frequency decomposition.

For instance, if no mask is present within a block, we only encode the mean value of the image. This obviously corresponds to the lowest frequency in the DCT. As another example, consider a block with a mask given by only the root node of a tree. This means that the inpainting only depends on the tonal values prescribed at the 4 corner points of the block. This means that there are very little fluctuations in the inpainting of that block. Hence, we can see this as the counterpart of the next higher frequency in the DCT. The more one descents in the tree, the better finer details can be represented, just as in the higher frequencies in the DCT.

In JPEG, coefficients corresponding to lower frequencies are quantised rather finely to obtain a more precise representation of the coarse image. In contrast, the coefficients for the higher frequencies are quantised much more coarsely as the human visual system is not very sensitive to errors here. The surprising similarity between the PDE-based compression approach and JPEG directly leads to the idea of adopting this adaptive quantisation. Thus, we make the quantisation dependent on the tree level and mimic the quantisation matrix of JPEG. To this end, we compute an individual quantisation parameter q at all mask points given by the formula

$$q = \operatorname{round}\left(\frac{Q}{\lambda^l}\right). \tag{6.4}$$

As in JPEG, the quality parameter $Q \in \{1, \dots, 256\}$ allows to steer the trade-off between the quality and the coding costs. Instead of the quantisation parameter q, we now store this quality parameter in the file header. Moreover, the quantisation parameter depends on the level l of the tree. In particular, we set $l = 0$ whenever there is no tree within a block. This means that the mean value is stored with a quantisation parameter of $q = Q$. The root node corresponds to the level $l = 1$ and is increased recursively by 1 for the children. A smaller value of l thus leads to a larger quantisation value, and vice versa. The fixed parameter $\lambda \geq 1$ determines the decay of the quantisation parameter from level to level. Experiments showed that $\lambda = 1.2$ gives good results.

Results

With the mask based on the binary subdivision tree and the adaptive quantisation, we perform the experiment with the new compression algorithm. Note that besides these two adaptations, everything else remains as described in Section 6.3.2. The corresponding results are depicted in Figure 6.22. Considering the compression quality in the left graph, one directly recognises that the difference between the different compression methods is rather small.

The PDE-based approach with the harmonic operator performs not very well for low compression ratios. With the biharmonic operator however, the compression quality is tremendously improved. The algorithm is able to outperform JPEG for all compression ratios larger than 40:1, and has the same performance for smaller compression ratios. This underlines the conceptual similarity of the two approaches. Due to the large improvement when using the biharmonic operator instead of the harmonic operator, the triharmonic operator is also tested. As it turns out, the resulting performance is almost identical to the one with the biharmonic operator. The higher sensitivity with respect to the quantisation apparently eliminates all benefits gained by the higher flexibility. As with the optimal masks, the PDE-based compression only surpasses JPEG2000 for very high compression ratios of at least 100:1.

For a visual impression, example results for a compression ratio of 55:1 are depicted in Figure 6.23. Also here, the PDE-based approach appears much more pleasant compared to the JPEG image. The main reason for this is that smooth transitions are better handled with the PDE-based approach, which leads to less differences at block boundaries in homogeneous areas. JPEG mainly represents the individual blocks in such smooth regions by the mean value, leading to unpleasant jumps between blocks. This effect is best visible at the right cheek. JPEG2000 overcomes the block artefacts by performing a discrete wavelet transform, which leads to a better overall impression.

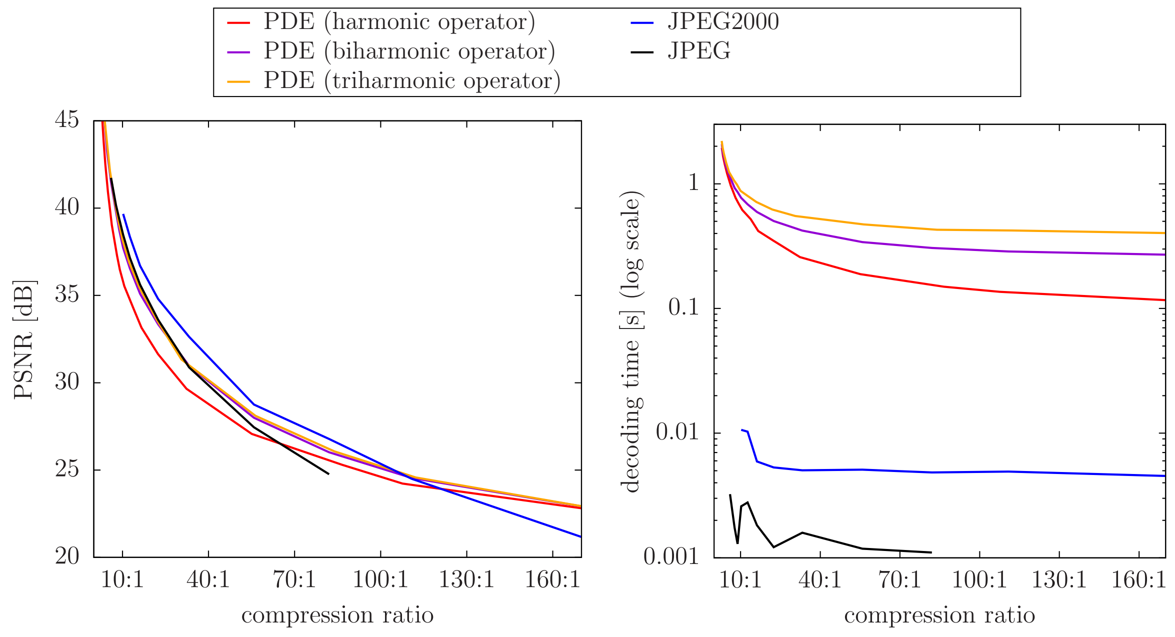

Figure 6.22. Compression results with block inpainting for different compression ratios for the image *trui*. Binary subdivision tree masks and an adaptive tree quantisation are used. **Left:** Compression performance comparison. **Right:** Comparison of decoding time.

However, the image seems to be more blurry, especially around the eyes.

Given the very promising results of the PDE-based compression, let us consider the resulting decoding times of the different methods shown in Figure 6.22 on the right. Unfortunately, despite the very fast processing of the blocks, the PDE-based approaches cannot achieve the very small runtime of JPEG or JPEG2000. This will be addressed in the following. Note the different decoding times for the three inpainting operators, which solely comes from the entropy coder. Apparently, higher order operators lead to less structured data which requires more effort to obtain an efficient entropy coding.

6.3.4 Accelerated Decoding

After having found a very good and fast PDE-based compression algorithm, we now want to focus on improving the decoding time. To this end, we will address two major bottlenecks in the following.

Entropy Coding

Despite its very good results, the PAQ entropy coder is very slow. JPEG and JPEG2000 rely on Huffman coding [96] and arithmetic coding [160, 161], respectively. Both entropy coders have a much better runtime performance compared to PAQ on

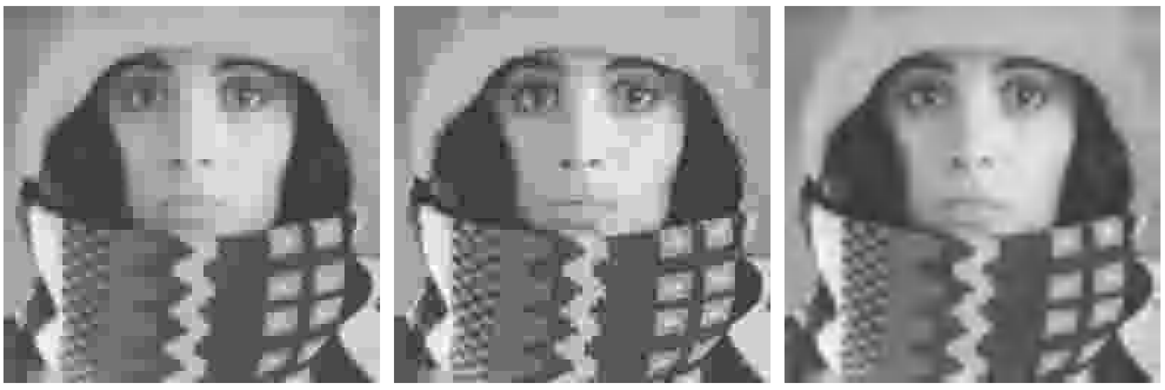

Figure 6.23. Example result of the PDE-based compression with subdivision tree strategies and comparable results of JPEG and JPEG2000 at the same compression ratio of 55:1.

the expense of being less efficient in terms of file size reduction. Thus, we replace the PAQ entropy coder by arithmetic coding or Huffman coding in our PDE-based compression.

Figure 6.24 shows the resulting compression performance. Both entropy coders lead to a tremendous speed up compared to the results with PAQ in Figure 6.22. For all compression ratios of approximately 30:1 and more, the PDE-based method is even faster than JPEG2000. The quality of the PDE-based compression suffers from the less efficient entropy coders. The best result is obtained with arithmetic coding and the biharmonic operator for all compression ratios. Due to the very similar runtime of both entropy coders, it makes sense to choose arithmetic coding as the favoured one.

Encoding Coefficients

A second bottleneck is the inpainting process itself: Solving the linear system of equations to obtain the coefficients of the discrete Green's functions requires some time, especially for larger mask densities. It is obvious from Figure 6.15 that this can last almost as long as the complete JPEG2000 decoding. Hence, we further speed up the algorithm by storing the coefficients and the mean value constant of the discrete Green's functions directly instead of the tonal values. This allows to obtain the inpainting within approximately 1ms independent of the operator or the mask density. As discussed in Section 3.4, these coefficients of the discrete Green's functions can also be considered as derivative information which is now prescribed at the mask points instead of the tonal information. Note that the PDE-based approach still needs to perform a cosine transform of the coefficients and an inverse transform of the modified values to obtain the reconstruction (cf. Algorithm 3.1). In contrast, JPEG only needs to perform the inverse transform of the already stored DCT coefficients. This means

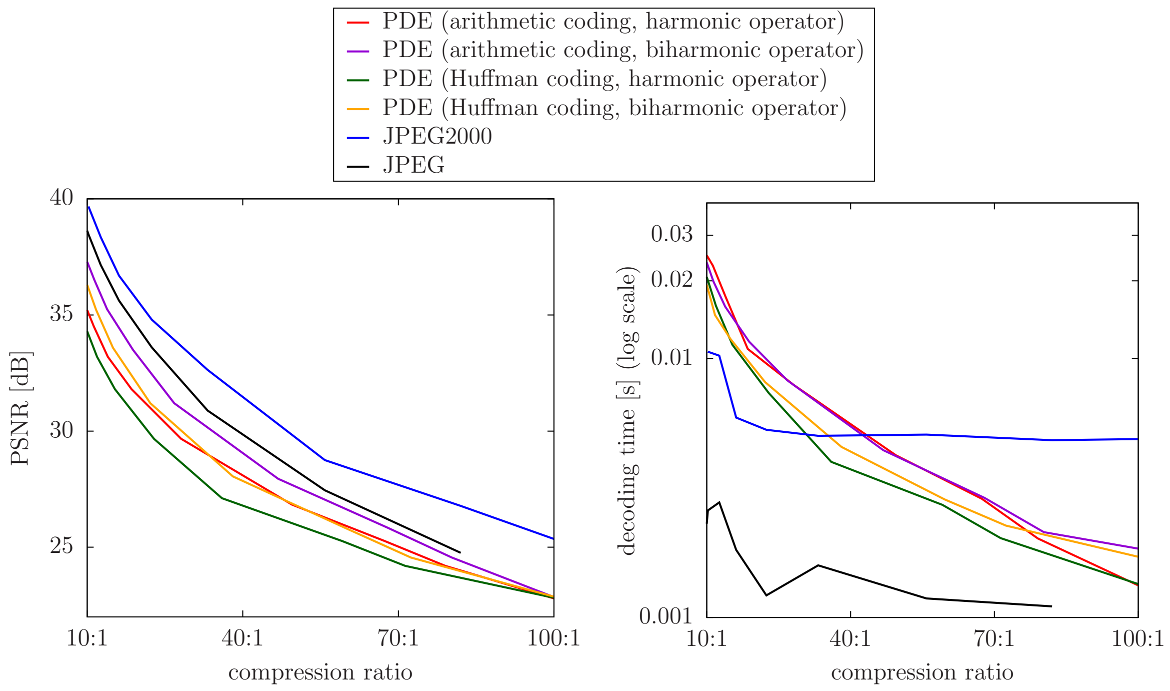

Figure 6.24. Accelerated compression results with different entropy coders for varying compression ratios for the image *trui*. Harmonic and biharmonic inpainting operators are tested.

that despite this enormous gain in time, we still have double the effort compared to JPEG.

For storing this new data efficiently, it has to be quantised, too. The mean value constant can be quantised in the same way as the tonal values. Unfortunately, the coefficients have a completely different range and scale compared to the tonal values: While the tonal values usually lie in $[0, 255]$, the coefficients lie within some interval $[-a, a]$, with some constant a depending on the inpainting operator. Thus, the coefficients are transferred to a different range by multiplying them with a certain constant $s \in (0, 2]$. Then, we apply a uniform quantisation on these values within the interval $[-127, 127]$. For reconstructing the coefficients, they are divided by s again. Besides the quality parameter, we also store the scale parameter s in a header with a precision of one byte. Both parameters are again optimised with a grid search to yield the best possible trade-off between file size and reconstruction quality. To find the best quantised coefficients for a specific quality and scale parameter, the quantisation-aware tonal optimisation method in Algorithm 4.6 is modified in a straightforward way. Thereby, besides adapting the quantisation itself, the inpainting echoes are replaced by the discrete Green's functions.

Similar to the experiment with the tonal values, we can check the influence of a quantisation on the coefficients. Again, the mask as well as the (quantised) coefficients

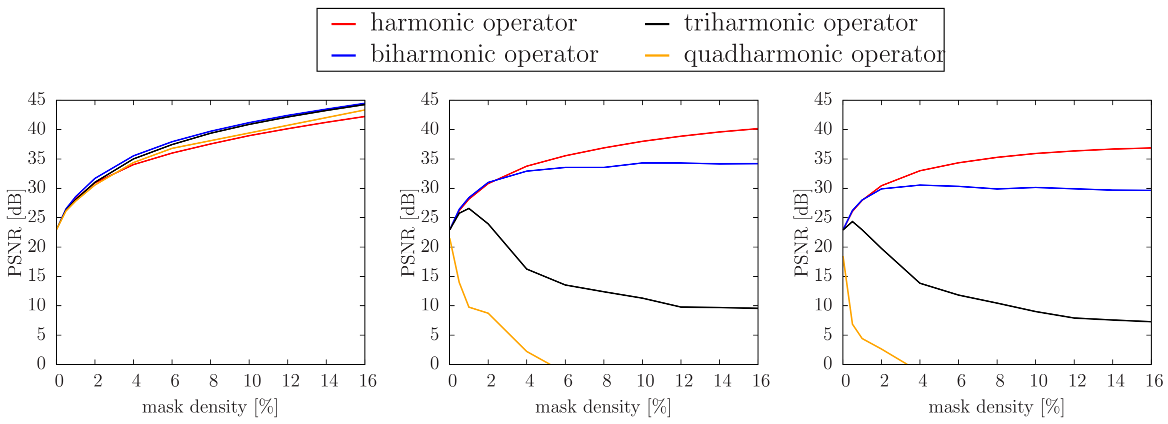

Figure 6.25. Comparison of best reconstructions from prescribed coefficients obtained with different operators and quantisation levels. **Left:** No quantisation. **Centre:** Quantisation with $q = 64$. **Right:** Quantisation with $q = 32$.

and constant are tuned to yield the best possible reconstruction for a given operator and quantisation value. The scaling parameter s is set to 1 which leaves the values unchanged. The result is shown in Figure 6.25. Recall that for continuous coefficients, the reconstruction is identical to the ones obtained from continuous tonal values. The quantisation of the coefficients has a much more severe influence on the reconstruction quality than a quantisation of the tonal values. This demonstrates that precision is very crucial when prescribing derivative information at mask locations, especially for higher order operators. In particular, note that the reconstructions even become worse when more points are present: An error in additional coefficients has a very bad influence on the inpainting result. This effect is even slightly visible for the biharmonic operator and $q = 32$, and much more severe for higher orders. Due to the very bad reconstructions given by the triharmonic or quadharmonic operators, we will only focus on the lower order operators when encoding coefficients.

In Figure 6.26, the new approach relying on the coefficients is compared to the previous compression results. Both rely on arithmetic coding as entropy coder. The reconstruction quality obtained by storing tonal values in combination with the biharmonic operator is the best among the PDE-based methods. Storing the coefficients leads to slightly worse results. The biharmonic operator also seems to be the best choice here, at least for smaller compression ratios. Considering the runtime, we have found a compression algorithm with a very small decoding time. We are able to outperform JPEG2000 for all compression ratios. Even though JPEG is still a bit more efficient, the maximum decoding time for the image trui is approximately 5ms, which is extremely fast. In total, one can say that any gain in time is on the cost of the quality.

To get a visual impression of the fast compression method, an example result is depicted in Figure 6.27. At a compression ratio of 55:1, the decoding time of 1.8ms

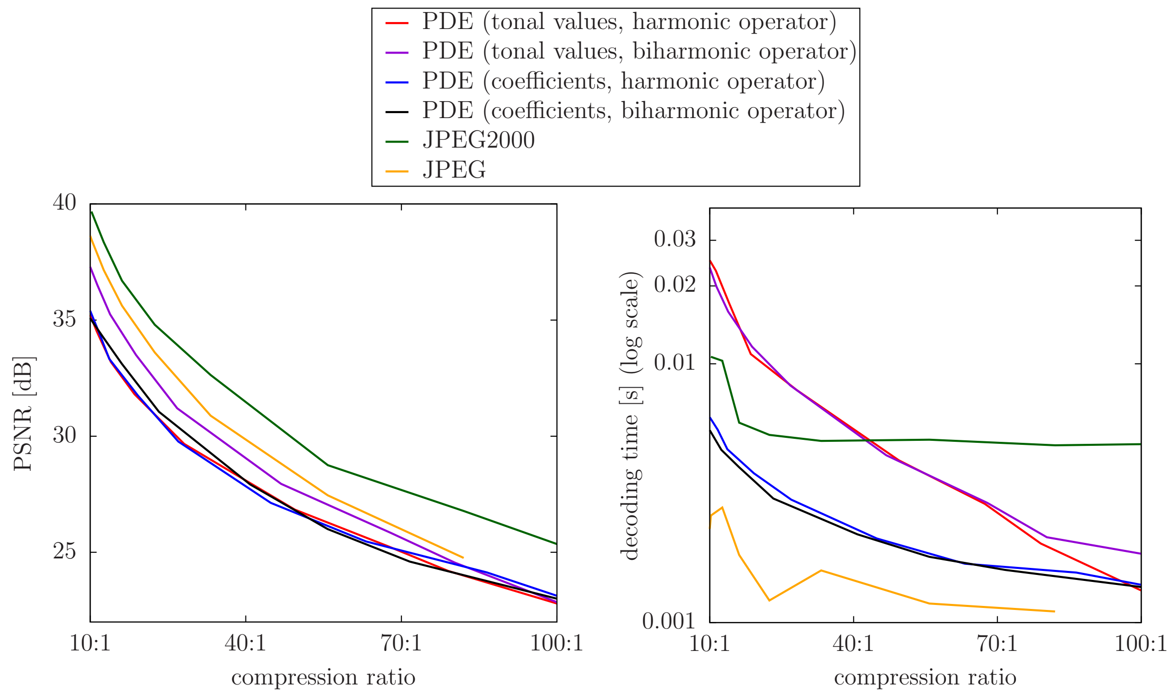

Figure 6.26. Comparison of compression with two different approaches for the image *trui*: Encoding coefficients of the discrete Green's functions on the one hand, and encoding tonal values on the other hand. Both variants are tested with harmonic and biharmonic inpainting, and rely on arithmetic coding.

is very fast. Similar to JPEG, the PDE-based approach shows block artefacts. Even though the JPEG image shows more texture, it suffers from drastic jumps between blocks, especially at the cheek. In contrast, the PDE-based approach has smoother transitions, but also shows less details. While both approaches yield images of comparable quality, the JPEG2000 image is clearly better.

6.3.5 Extension to Colour Images

In a final step, we want to extend the PDE-based compression algorithm such that it can also handle colour images. To this end, the image is initially transformed from the RGB colour space into the YCbCr colour space. This idea has already been successfully used in related compression methods such as JPEG or even PDE-based approaches relying on EED-inpainting [154, 152]. It exploits the following fact: The human visual system is much more sensitive to the luma channel Y compared to the chroma channels Cb and Cr. Thus, it is possible to allow for a lower quality in the Cb and Cr channels, while encoding the Y channel in a higher quality.

In our compression algorithm, we also mainly consider the Y channel. As before, we first optimise the spatial and tonal data to obtain the best reconstruction for only this

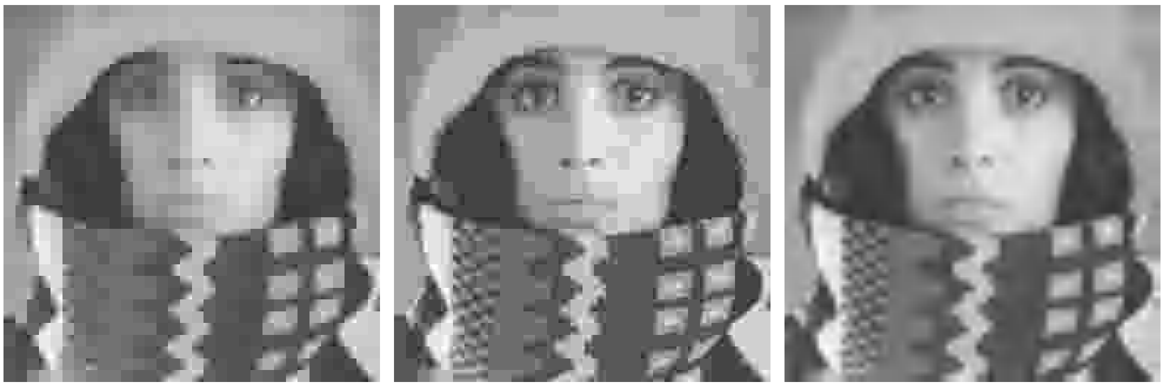

Figure 6.27. Example result of the PDE-based compression, using coefficients to be stored, arithmetic coding and biharmonic operator. The decoding time is 1.8ms. Comparable results of JPEG and JPEG2000 at the same compression ratio of 55:1 are shown, too.

specific channel. Then, this mask is reused for the chroma channels as well, because all structural information is already represented in the Y channel. This allows to store only one mask for all three channels which is very efficient. Moreover, we follow the idea of JPEG and store the chroma values much coarser than the luma values. This is achieved by always using only half as many quantisation levels in these two channels than in the Y channel. In particular, this channel dependent quantisation is also integrated into the process of finding optimal tonal data. Note that for all optimisation steps, the error is considered in the YCbCr space. In the decoding step, the individual channels are inpainted separately as before, and the final image is given by a back-transformation from the YCbCr colour space to the RGB colour space.

With this extension to colour images, we perform some experiments with the very fast PDE-based image compression including the arithmetic coding and direct encoding of coefficients. For the image *peppers* in Figure 6.28, the fast PDE-compression is able to beat JPEG for a compression ratio of 7:1. This is a remarkable result as we did not mainly aim to obtain the best quality. We rather prioritised the method which is very efficient in terms of decoding speed. Therefore, the decoding with the PDE approach on the 512×512 colour image only takes 80ms.

More results on different images and compression ratios are shown in Figures 6.29 and 6.30. Here, the JPEG image achieves a higher quality in terms of the PNSR error measure. These images contain more texture compared to the peppers images which are easier to handle for JPEG. Nevertheless, the PDE approach is still able to yield a convincing image with no major details missing.

Thanks to the very fast processing, we are even able to process a very large HD image of size 2560×1920, see Figure 6.31. The decoding of the image took only

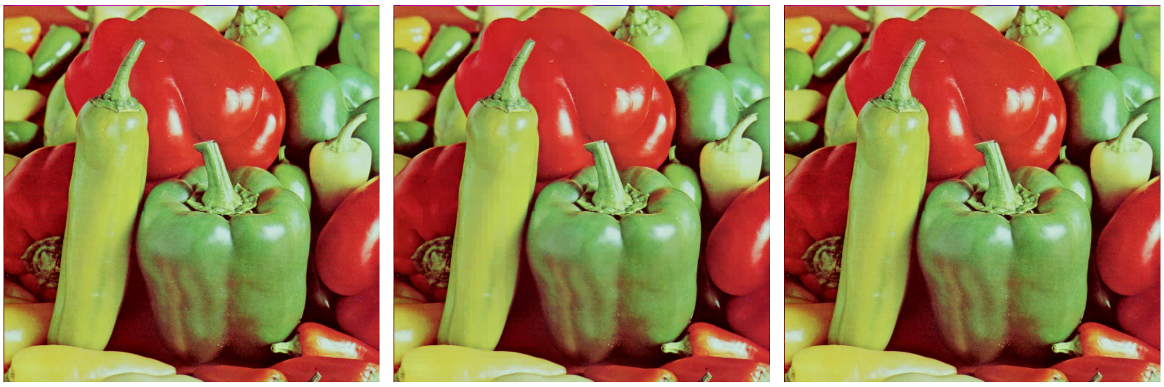

Figure 6.28. Example with a compression ratio of 7:1. **Left:** Original image *peppers* (size: 512×512). **Centre:** PDE approach (PSNR: 33.40 dB). **Right:** JPEG (PSNR: 32.93 dB).

0.81 seconds. For a comparison, JPEG needs around 0.24s and JPEG2000 even takes 0.93s, so the PDE runtime is very good. This shows that the presented approach is also capable to process very large colour images efficiently, even though it incorporates solving partial differential equations. This has never been achieved before on a CPU. Despite the worse PSNR value of the PDE approach, one cannot see a large visual difference to JPEG or JPEG2000. In particular, the overall impression of all three methods compared to the original image is very positive.

6.3.6 Conclusion and Outlook

In this Section, we have seen that linear PDEs are a powerful tool for obtaining fast and high quality compression algorithms. The key for success is a clever combination and tuning of the individual ingredients: The block subdivision allows a fast inpainting and associated data optimisation, which yields very good reconstructions even with linear PDEs. Thereby, the theory of discrete Green's functions allows a fast and accurate processing. Together with efficient encoding strategies such as tree subdivision masks, fast entropy coders or processing in a different colour space, we obtain a very good compression algorithm. Despite its similarity to JPEG in various regards, the underlying inpainting process as a core ingredient represents a major conceptual difference. In total, it could be demonstrated for the first time that a PDE based approach is able to compete with JPEG in terms of both quality and decoding time.

In the future, a more efficient entropy coding for the extracted spatial and tonal data could help to boost the compression performance. Even though the fast arithmetic coding already yields good results, the experiments with the slower PAQ showed that there is still a lot of potential. In particular, a designated adaption to the specific structure of the data could help to reduce the final file size.

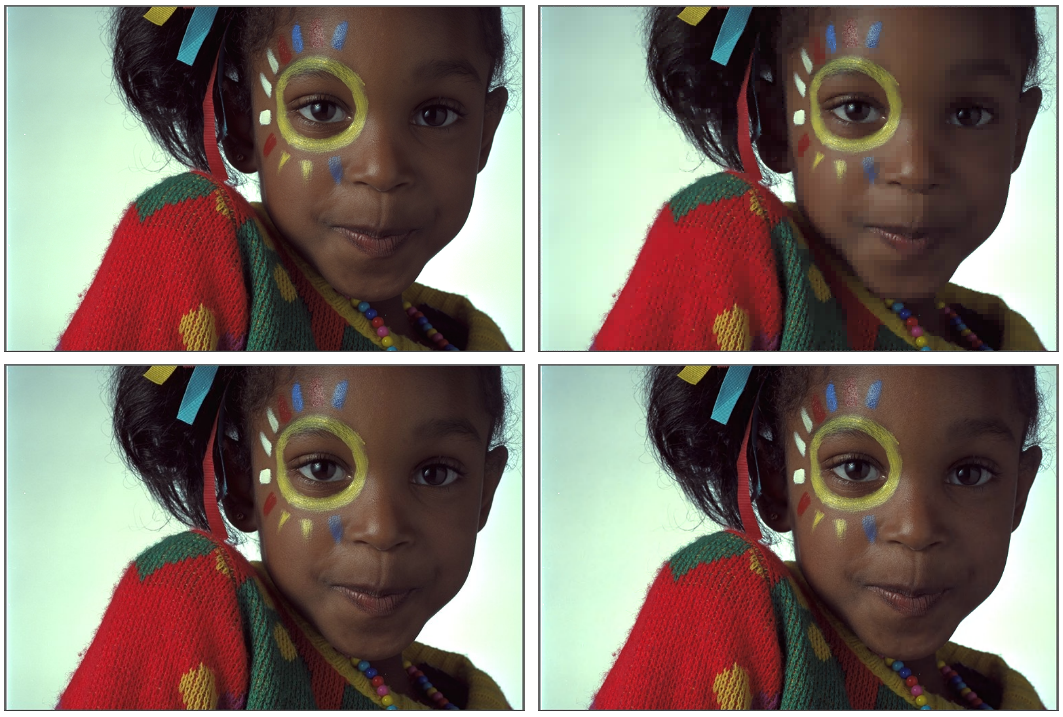

Figure 6.29. Example with a compression ratio of 13:1. **Top left:** Original image *kodim15* from the Kodak image database [56] (size: 768 × 512). **Top right:** PDE approach (PSNR: 33.02 dB). **Bottom left:** JPEG (PSNR: 37.88 dB). **Bottom right:** JPEG2000 (PSNR: 41.02 dB).

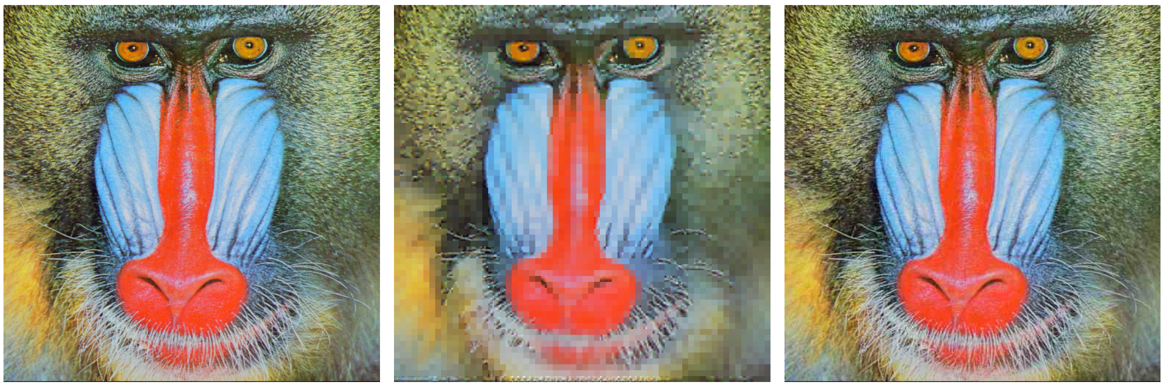

Figure 6.30. Example with a compression ratio of 25:1. **Left:** Original image *baboon* (size: 512 × 512). **Centre:** PDE approach (PSNR: 20.36 dB). **Right:** JPEG (PSNR: 23.58 dB).

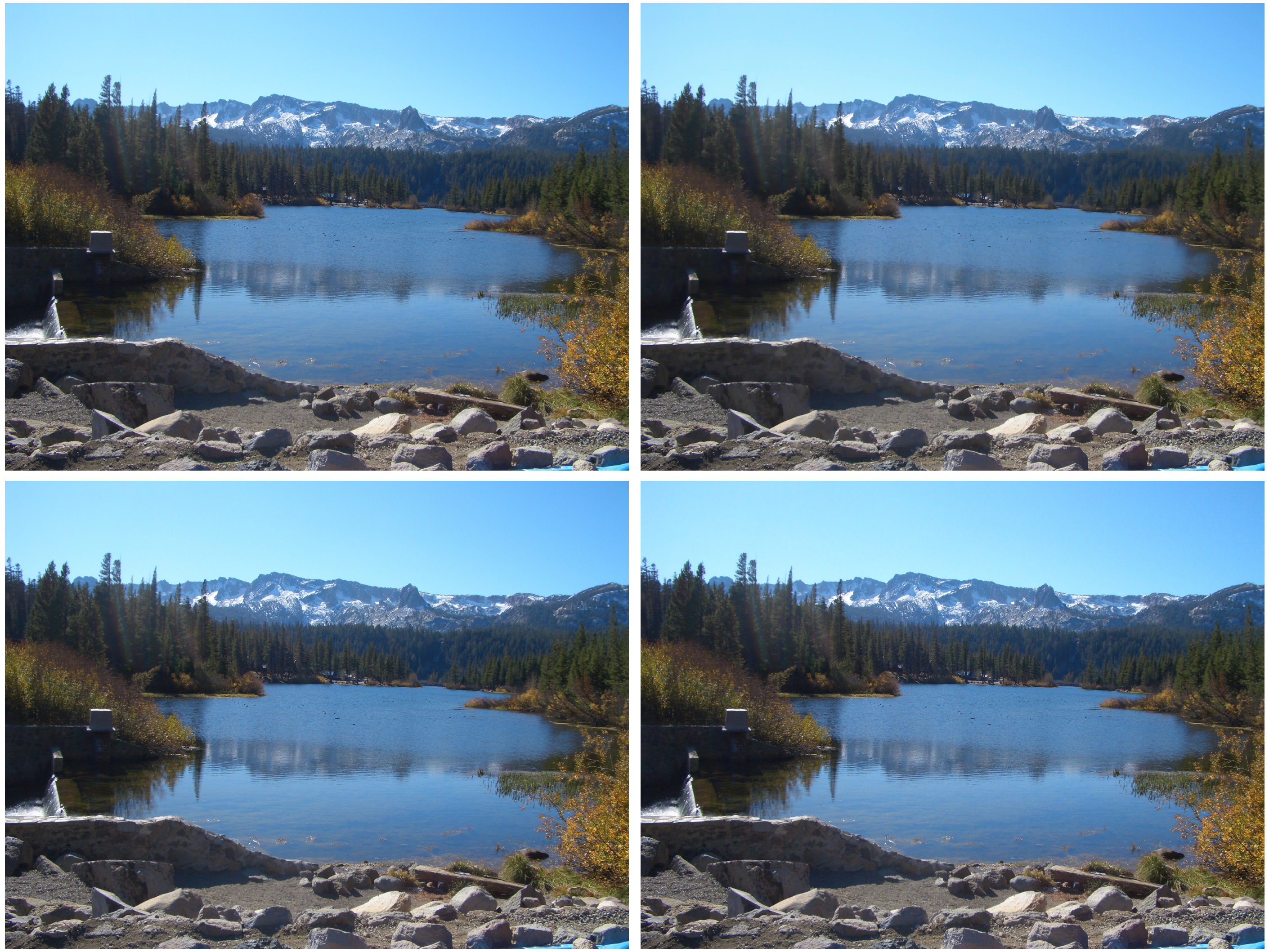

Figure 6.31. Example with a compression ratio of 15:1. **Top left:** Original image *lake* (size: 2560×1920). **Top right:** PDE approach (PSNR: 34.20 dB). **Bottom left:** JPEG (38.04 dB). **Bottom right:** JPEG2000 (40.82 dB).

Chapter 7

Conclusion and Outlook

7.1 Summary and Conclusion

In the present thesis, we have analysed the potential of linear partial differential equations for image compression. Chapter 1 started by giving a general introduction to image compression. Then, the general idea of PDE-based inpainting and its usage in a compression context have been presented. Further, there has been an overview over related methods in the respective fields.

In Chapter 2, the inpainting problem itself has been discussed. Based on diffusion processes, we have introduced the standard continuous formulation of PDE-based inpainting. The corresponding discrete analogue has been presented and analysed afterwards. On the one hand, we have considered general known properties of the inpainting solution such as the maximum-minimum principle for homogeneous diffusion inpainting. On the other hand, we have seen that the condition number and consequently the complexity of solving the inpainting problem using the standard formulation is typically inversely proportional to the mask density. In particular when there are big gaps between mask points, filling in information in the larger unknown areas becomes expensive. This explains the success of bidirectional multigrid methods to solve the inpainting problem as they reduce the distance between mask points on coarser levels.

With the help of discrete Green's functions, we were able to reformulate the original inpainting formulation in terms of sparsity in Chapter 3. This allowed to rewrite the inpainting solution as superposition of atomic components given by the discrete Green's functions. Besides gaining theoretical insights, we have seen that this theory also led to many practical advantages. Compared to the standard formulation of the inpainting problem, we can consider this reformulated problem as a dual problem: Operations become less expensive the less mask points are present, which is the opposite to the standard inpainting. In this duality, discrete Green's functions can be related to inpainting echoes.

The tonal optimisation problem is considered in Chapter 4. While there is a unique solution of the problem for linear operators, we have seen that the fastest method highly depends on the mask density and the order of the operator. Again, the duality of the two inpainting formulations becomes apparent: While the approach relying on discrete Green's functions allows a fast processing for low mask densities, the gradient descent approach using the standard formulation is faster for larger mask densities. The primal-dual method is a very fast alternative for the harmonic operator and denser masks. Moreover, a quantisation-aware tonal optimisation is possible with inpainting echoes, and a gradient descent scheme is even able to cope with nonlinear inpainting operators.

Chapter 5 discussed the spatial optimisation in detail. We considered different methods, and found that again different methods are suited for specific scenarios. When time is not an issue, the best results for the linear operators are obtained with the optimal control approach. However, it is the only method where the mask density cannot be specified explicitly. The analytic approach relying on the theory of Belhachmi *et al.* is a promising fast alternative for homogeneous diffusion inpainting with slightly worse results. A very good trade-off between runtime and quality is given by the gradient descent with discrete Green's function. It allows to find a very good masks with the harmonic operator for discrete and even for continuous mask locations. The probabilistic methods are able to yield reliable results for arbitrary operators, in particular for nonlinear operators.

The actual performance of PDE-based inpainting with linear operators within a compression context is evaluated in Chapter 6. The first method uses a segment-based inpainting for the purpose of depth map compression. By combining a mask on a regular hexagonal structure with optimised free mask points, very good compression results could be achieved. Due to the dedicated design for this type of images, the codec was able to outperform JPEG and JPEG2000, while competing with HEVC. The second approach encodes the highly optimised masks from the optimal control approach and achieves remarkable results on general images. Even with the linear PDEs it is possible to outperform JPEG, JPEG2000 or R-EED. The third compression approach demonstrated that PDE-based image compression can not only convince qualitatively, but can also be very fast when it comes to the decoding. The key to success was to use a block-based inpainting which makes all operations very efficient. This allowed to incorporate a tonal optimisation into the spatial optimisation process, leading to very good inpainting data. By encoding the coefficients of the discrete Green's functions, the decoding turned out to be very fast. In the end, we were able to compete with JPEG in terms of quality and decoding time.

To summarise, we could demonstrate that linear PDEs are a very powerful tool for image compression. The high quality reconstructions do not only allow to compete with related compression approaches in terms of quality. Against all prejudices, the linear operators even allow to be in the same league as JPEG or JPEG2000 when it comes to decoding time. To reach this goal, we exploited the derived insights on the inpainting process, in particular the duality between the standard and reformulated inpainting formulation using discrete Green's functions. This demonstrates that it always pays off to put effort into the analysis and understanding of the underlying method.

We have seen that there are several crucial ingredients to obtain a successful codec with PDEs. First of all, it is important to incorporate as much a priori information of the images as possible. For instance, we did this when using the segment-based inpainting to recover the characteristic sharp edges and smooth transitions of the depth maps. Moreover, it makes sense to optimise the inpainting data as good as possible on the encoding side, while at the same time including the final entropy coding into the optimisation process. To obtain a fast decoding, a localisation along with fast algorithmic realisations is very important. In particular, it makes sense to make certain sacrifices in the quality to obtain a faster decoding as we have done it by encoding the coefficients directly instead of the tonal data.

7.2 Outlook

Despite its maturity, it is clear that PDE-based compression has not reached a final stage yet. There is still room for improvement in various aspects.

In Section 6.3.1, we have seen that it pays off to jointly optimise the tonal and spatial data. Moreover, we have seen in Section 4.9 that incorporating the quantisation into the tonal optimisation process is better compared to a subsequent quantisation of the continuous optimal values. In the PDE-based image compression, we have many more dependencies. For instance, depending on the entropy coder, one could think about choosing the mask points more likely at those positions which can be encoded more efficiently. Clearly, the quantisation also has an influence on the chosen mask points. This suggests to have a joint optimisation of all ingredients of the compression algorithm.

In this context, a better understanding of the entropy coding is crucial. Ideally, we would know the actual coding costs of the individual components such as the mask points, segment boundaries or the quantisation levels. This could then be incorporated into a variational framework, where an appropriate energy functional is minimised to find the best ratio between the ingredients. In particular, this would avoid the need to perform brute force search strategies as we have done it for the quantisation parameter in Section 6.2.1.

In contrast to encoding the tonal values, we have seen in Section 6.3.4 that it is also possible to encode higher order derivative information. In [36], it is demonstrated that it is possible to obtain very good reconstructions from sparse gradient information. Even though the reconstruction quality is very good, we have recently shown that this strategy is not perfectly suited for image compression [8]. The main reason for this is the high sensitivity of the gradient data to a quantisation. In the future, an improved entropy coder could potentially allow to store this data in a higher precision and lead to better compression results. Moreover, one could think of using even other kinds of sparse features such as the diffusivity information or to reconstruct an image from a combination of different features.

Of course, an improved and faster encoding is desired in the future. To this end, one could for example try to find fast alternatives for the discussed data optimisation strategies. Especially when it comes to finding the mask, faster methods would clearly help PDE-based codecs to further advance. Perhaps, making small sacrifices on the quality would even lead to tremendous accelerations, which in turn would allow for compression of large amounts of data. In this context, a fast GPU implementation of the tonal optimisation with discrete Green's functions could help to speed up the computations. Moreover, improved entropy coders which are specifically designed for the purpose of PDE-based image compression could help to further improve the coding performance.

The theoretical insights on the inpainting with the help of the Green's functions can also be extended. In [7], we have already considered one-dimensional discrete Green's functions for sparse signal approximation. We have shown that the theory is also valid for this setting, and derived simple closed-form expressions for the 1D discrete Green's functions. In the future, one can think of considering discrete Green's functions in higher dimensions or even for nonlinear operators. Perhaps, this would enable to find a connection between EED inpainting and sparsity.

As demonstrated in [153], PDE-based methods are perfectly suited fo progressive modes. Also the presented compression algorithms within this thesis can potentially be extended to support this feature. In particular, we could for instance first select the inpainting data such that the coarse image structure is represented. Then, more and more points are added to include detail information, which is successively transferred to the recipient. In particular, the discrete Green's functions are very well suited for this task: Adding additional points simply means adding further discrete Green's functions. In particular, there is no need to perform a complete inpainting after sending more data.

Clearly, one can employ or adapt the presented methods for any type of data. For example, the efficient storage of MRI images is still an open issue in medical imaging, and encoding high-dynamic range (HDR) images is becoming more and more important.

Besides all the other work in the field of PDE-based image compression, this thesis is another small step towards a unified and powerful PDE-based codec, which will maybe succeed to be a future standard for image compression tasks someday.

Bibliography

Own Publications

[1] L. Hoeltgen, M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, S. Setzer, D. Johannsen, F. Neumann, and B. Doerr. Optimising spatial and tonal data for PDE-based inpainting. In M. Bergounioux, G. Peyré, C. Schnörr, J.-P. Caillau, and T. Haberkorn, editors, *Variational Methods in Imaging and Geometric Control*, pages 35–83, De Gruyter, Berlin, 2017.

[2] S. Hoffmann, M. Mainberger, J. Weickert, and M. Puhl. Compression of depth maps with segment-based homogeneous diffusion. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 319–330. Springer, Berlin, 2013.

[3] S. Hoffmann, G. Plonka, and J. Weickert. Discrete Green's functions for harmonic and biharmonic inpainting with sparse atoms. In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 8932 of *Lecture Notes in Computer Science*, pages 169–182. Springer, Berlin, 2015.

[4] M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, D. Johannsen, F. Neumann, and B. Doerr. Optimising spatial and tonal data for homogeneous diffusion inpainting. In A. M. Bruckstein, B. ter Haar Romeny, A. M. Bronstein, and M. M. Bronstein, editors, *Scale Space and Variational Methods in Computer Vision*, volume 6667 of *Lecture Notes in Computer Science*, pages 26–37. Springer, Berlin, 2012.

[5] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert. From optimised inpainting with linear PDEs towards competitive image compression codecs. In M. Rivera X. Yu T. Bräunl, B. McCane, editors, *Image and Video Technology*, volume 9431 of *Lecture Notes in Computer Science*, pages 63–74, Cham, April 2016. Springer.

[6] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert. Evaluating the true potential of diffusion-based inpainting in a compression context. *Signal Processing: Image Communication*, 46:40–53, 2016.

[7] G. Plonka, S. Hoffmann, and J. Weickert. Pseudo-inverses of difference matrices

and their application to sparse signal approximation. In *Linear Algebra and its Applications*, volume 503, pages 26–47, August 2016.

[8] M. Schneider, P. Peter, S. Hoffmann, J. Weickert, and E. Meinhardt-Llopis. Gradients versus grey values for sparse image reconstruction and inpainting-based compression. In J. Blanc-Talon, C. Distante, W. Philips, D. Popescu, and P. Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 10016 of *Lecture Notes in Computer Science*, pages 1–13, Cham, 2016. Springer.

General References

[9] T. Acar and M. Gökmen. Image coding using weak membrane model of images. In A. K. Katsaggelos, editor, *Visual Communications and Image Processing '94*, volume 2308 of *Proceedings of SPIE*, pages 1221–1230. SPIE Press, Bellingham, 1994.

[10] A. Alahmar. Recursive PDE-based compression of high-dimensional data sets. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[11] G. Allaire, K. Trabelsi, and S.M. Kaber. *Numerical Linear Algebra.* Texts in Applied Mathematics. Springer, New York, 2008.

[12] F. Alter, S. Durand, and J. Froment. Adapted total variation for artifact free decompression of JPEG images. *Journal of Mathematical Imaging and Vision*, 23(2):199–211, September 2005.

[13] H. A. Aly and E. Dubois. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing*, 14(10):1647–1659, October 2005.

[14] M. Arigovindan, M. Sühling, P. Hunziker, and M. Unser. Variational image reconstruction from arbitraily spaced samples: A fast multiresolution spline solution. *IEEE Transactions on Image Processing*, 14(4):450–460, April 2005.

[15] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*, volume 147 of *Applied Mathematical Sciences.* Springer, New York, 2002.

[16] V. Aurich and U. Daub. Bilddatenkompression mit geplanten Verlusten und hoher Rate. In B. Jähne, P. Geißler, H. Haußecker, and F. Hering, editors, *Mustererkennung 1996*, pages 138–146. Springer, Berlin, 1996.

[17] E. Bae and J. Weickert. Partial differential equations for interpolation and compression of surfaces. In M. Dæhlen, M. Floater, T. Lyche, J.-L. Merrien, K. Mørken, and L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces*, volume 5862 of *Lecture Notes in Computer Science*, pages 1–14. Springer, Berlin, 2010.

[18] R. Bajcsy and S. Kovačič. Multiresolution elastic matching. *Computer Vision, Graphics and Image Processing*, 46(1):1–21, April 1989.

[19] V. Bastani, M. S. Helfroush, and K. Kasiri. Image compression based on spatial redundancy removal and image inpainting. *Journal of Zhejiang University – Science C (Computers & Electronics)*, 11(2):92–100, 2010.

[20] S. Battiato, G. Gallo, and F. Stanco. Smart interpolation by anisotropic diffusion. In *Proc. Twelvth International Conference on Image Analysis and Processing*, pages 572–577, Montova, Italy, September 2003. IEEE Computer Society Press.

[21] K. Baum. GPU supported diffusion-based naive video compression. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[22] A. Belahmidi and F. Guichard. A partial differential equation approach to image zoom. In *Proc. 2004 IEEE International Conference on Image Processing*, volume 1, pages 649–652, Singapore, October 2004.

[23] Z. Belhachmi, D. Bucur, B. Burgeth, and J. Weickert. How to choose interpolation data in images. *SIAM Journal on Applied Mathematics*, 70(1):333–352, 2009.

[24] J. M. Berger and G. J. Lasher. The use of discrete Green's functions in the numerical solution of Poisson's equation. *Illinois Journal of Mathematics*, 2(4A):593–607, November 1958.

[25] M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. SIGGRAPH 2000*, pages 417–424, New Orleans, LI, July 2000.

[26] F. Bornemann and T. März. Fast image inpainting based on coherence transport. *Journal of Mathematical Imaging and Vision*, 28(3):259–278, July 2007.

[27] A. Borzì, H. Grossauer, and O. Scherzer. Analysis of iterative methods for solving a ginzburg-landau equation. *International Journal of Computer Vision*, 64(2–3):203–219, 2005.

[28] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio, and Y. LeCun. High quality document image compression with DjVu. *Journal of Electronic Imaging*, 7(3):410–425, July 1998.

[29] S. Bougleux, G. Peyré, and L. Cohen. Image compression with anisotropic triangulations. In *Proc. Tenth International Conference on Computer Vision*, pages 2343–2348, Kyoto, Japan, October 2009.

[30] A. Bourquard and M. Unser. Anisotropic interpolation of sparse generalized image samples. *IEEE Transactions on Image Processing*, 22(2):459–472, 2013.

[31] S.P. Boyd and L. Vandenberghe. *Convex Optimization.* Berichte über verteilte Messysteme. Cambridge University Press, 2004.

[32] R. N. Bracewell. *The Fourier Transform and its Applications.* McGraw-Hill, New York, 1999.

[33] K. Bredies and M. Holler. A total variation-based JPEG decompression model. *SIAM Journal on Imaging Sciences*, 5(1):366–393, 2012.

[34] K. Bredies and M. Holler. A tgv regularized wavelet based zooming model. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 149–160. Springer, Berlin, 2013.

[35] W. L. Briggs. *A Multigrid Tutorial.* SIAM, Philadelphia, 1987.

[36] E.-M. Brinkmann, M. Burger, and I. Grah. Regularization with sparse vector fields: From image compression to TV-type reconstruction. In J.-F. Aujol, M. Nikolova, and N. Papadakis, editors, *Scale Space and Variational Methods in Computer Vision*, volume 9087 of *Lecture Notes in Computer Science*, pages 191–202. Springer, Berlin, 2015.

[37] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In T. Pajdla and J. Matas, editors, *Computer Vision – ECCV 2004, Part IV*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36. Springer, Berlin, 2004.

[38] M. D. Buhmann. *Radial Basis Functions.* Cambridge University Press, Cambridge, UK, 2003.

[39] S. Carlsson. Sketch based coding of grey level images. *Signal Processing*, 15:57–83, 1988.

[40] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH 2001, pages 67–76, New York, 2001.

[41] V. Caselles, J.-M. Morel, and C. Sbert. An axiomatic approach to image interpolation. *IEEE Transactions on Image Processing*, 7(3):376–386, March 1998.

[42] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[43] T. F. Chan and J. Shen. Non-texture inpainting by curvature-driven diffusions (CDD). *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.

[44] T. F. Chan and J. Shen. *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods.* SIAM, Philadelphia, 2005.

[45] T. F. Chan and H. M. Zhou. Feature preserving lossy image compression using nonlinear PDE's. In F. T. Luk, editor, *Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, volume 3461 of *Proceedings of SPIE*, pages 316–327. SPIE Press, Bellingham, 1998.

[46] T. F. Chan and H. M. Zhou. Total variation improved wavelet thresholding in image compression. In *Proc. Seventh International Conference on Image Processing*, volume II, pages 391–394, Vancouver, Canada, September 2000.

[47] H.-C. Chang and L.-C. Wang. A simple proof of Thue's theorem on circle packing. *ArXiv e-prints*, September 2010.

[48] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proc. 1994 IEEE International Conference on Image Processing*, volume 2, pages 168–172, Austin, TX, November 1994. IEEE Computer Society Press.

[49] J. Chen, F. Ye, J. Di, C. Liu, and A. Men. Depth map compression via edge-based inpainting. In *Proc. 2012 Picture Coding Symposium*, pages 57–60, Kraków, Poland, May 2012.

[50] J. Chen, F. Ye, J. Di, C. Liu, and A. Men. Depth map compression via edge-based inpainting. In *Picture Coding Symposium*, pages 57–60, Kraków, Poland, May 2012.

[51] Y. Chen, R. Ranftl, and T. Pock. A bi-level view of inpainting-based image compression. In Z. Kúkelová and J. Heller, editors, *Proc. 19th Computer Vision Winter Workshop*, Křtiny, Czech Republic, February 2014.

[52] F. Chung and S.-T. Yau. Discrete Green's functions. *Journal of Combinatorial Theory, Series A*, 91(1–2):191–214, 2000.

[53] P. G. Ciarlet and J.-L. Lions, editors. *Handbook of Numerical Analysis. Volume II – Finite Element Methods (Part 1).* North Holland, Amsterdam, 1991.

[54] A. Cohen, N. Dyn, F. Hecht, and J.-M. Mirebeau. Adaptive multiresolution analysis based on anisotropic triangulations. *Mathematics of Computation*, 81(278):789–810, 2012.

[55] D. Colton. *Partial Differential Equations: An Introduction.* Dover, New York, 2004.

[56] Eastman Kodak Company. Kodak true color image suite, 1999. Available online: `http://r0k.us/graphics/kodak/`.

[57] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965.

[58] S. Cuomo, A. Galletti, G. Giunta, and A. Starace. Surface reconstruction from scattered point via RBF interpolation on GPU. In *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on*, pages 433–440, September 2013.

[59] L. Demaret, N. Dyn, and A. Iske. Image compression by linear splines over adaptive triangulations. *Signal Processing*, 86(7):1604–1616, 2006.

[60] U. Y. Desai, M. M. Mizuki, I. Masaki, and B. K. P. Horn. Edge and mean based image compression. Technical Report 1584 (A.I. Memo), Artificial Intelligence Lab., Massachusetts Institute of Technology, Cambridge, MA, November 1996.

[61] G. Di Blasi, E. Francomano, A. Tortorici, and E. Toscano. A smoothed particle image reconstruction method. *Calcolo*, 48(1):61–74, 2011.

[62] R. Distasi, M. Nappi, and S. Vitulano. Image compression by B-tree triangular coding. *IEEE Transactions on Communications*, 45(9):1095–1100, September 1997.

[63] J. Duchon. Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. *RAIRO Analyse Numérique*, 10(3):5–12, 1976.

[64] S. Durand and M. Nikolova. Restoration of wavelet coefficients by minimizing a specially designed objective function. In O. Faugeras and N. Paragios, editors, *Proc. Second IEEE Workshop on Geometric and Level Set Methods in Computer Vision*, Nice, France, October 2003. INRIA.

[65] N. Dyn, D. Levin, and S. Rippa. Data dependent triangulations for piecewise linear interpolation. *IMA Journal of Numerical Analysis*, 10(1):137–154, 1990.

[66] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038, Corfu, September 1999.

[67] J. H. Elder. Are edges incomplete? *International Journal of Computer Vision*, 34(2/3):97–122, 1999.

[68] G. Evans, J.M. Blackledge, and P. Yardley. *Analytic Methods for Partial Differential Equations*. Springer, London, 2000.

[69] G. Facciolo, P. Arias, V. Caselles, and G. Sapiro. Exemplar-based interpolation of sparsely sampled images. In D. Cremers, Y. Boykov, A. Blake, and F. R. Schmidt, editors, *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, volume 5681 of *Lecture Notes in Computer Science*, pages 331–344. Springer, Berlin, 2009.

[70] G. Facciolo, F. Lecumberry, A. Almansa, A. Pardo, V. Caselles, and B. Rougé. Constrained anisotropic diffusion and some applications. In *Proc. 2006 British Machine Vision Conference*, volume 3, pages 1049–1058, Edinburgh, Scotland, September 2006.

[71] F. Faille and M. Petrou. Invariant image reconstruction from irregular samples and hexagonal grid splines. *Image and Vision Computing*, 28(8):1173–1183, August 2010.

[72] D. S. Feingold and R. S. Varga. Block diagonally dominant matrices and generalizations of the gerschgorin circle theorem. *Pacific Journal of Mathematics 12*, pages 1241–1250, 1962.

[73] R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial grey scale. *Proceedings of the Society of Information Display*, 17:75–77, 1976.

[74] G. E. Ford. Application of inhomogeneous diffusion to image and video coding. In *Proc. 13th Asilomar Conference on Signals, Systems and Computers*, volume 2, pages 926–930, Asilomar, CA, November 1996.

[75] G. E. Ford, R. R. Estes, and H. Chen. Scale-space analysis for image sampling and interpolation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 165–168, San Francisco, CA, March 1992.

[76] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38:181–200, 1982.

[77] P. Fränti and O. Nevalainen. Compression of binary images by composite methods based on block coding. *Journal of Visual Communication and Image Representation*, 6(4):366–377, December 1995.

[78] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, and H.-P. Seidel. Towards PDE-based image compression. In N. Paragios, O. Faugeras, T. Chan, and C. Schnörr, editors, *Variational, Geometric and Level-Set Methods in Computer Vision*, volume 3752 of *Lecture Notes in Computer Science*, pages 37–48. Springer, Berlin, 2005.

[79] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, and H.-P. Seidel. Image compression with anisotropic diffusion. *Journal of Mathematical Imaging and Vision*, 31(2–3):255–269, July 2008.

[80] J. Gautier, O. Le Meur, and C. Guillemot. Efficient depth map compression based on lossless edge coding and diffusion. In *Proc. 2012 Picture Coding Symposium*, pages 81–84, Kraków, Poland, May 2012.

[81] J. Gautier, O. Le Meur, and C. Guillemot. Efficient depth map compression based on lossless edge coding and diffusion. In *Picture Coding Symposium*, pages 81–84, Kraków, Poland, May 2012.

[82] M. Gay, A. Lampe, and M. Breiling. Implementation of the OMP algorithm for sparse recovery tasks in ODFM systems. *8th Karlsruhe Workshop on Software Radios*, pages 117–126, March 2014.

[83] G. H. Golub and C. F. Van Loan. *Matrix computations*. The John Hopkins University Press, New York, 1996.

[84] R. Gomathi and A. V. A. Kumar. A multiresolution image completion algorithm for compressing digital color images. *Journal of Applied Mathematics*, 2014:Article ID 757318, 2014.

[85] A. Gothandaraman, R. Whitaker, and J. Gregor. Total variation for the removal of blocking effects in DCT based encoding. In *Proc. 2001 IEEE International Conference on Image Processing*, volume 2, pages 455–458, Thessaloniki, Greece, October 2001.

[86] L. J. Grady and J. Polimeni. *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. Springer, Berlin, 2010.

[87] S. Grewenig, J. Weickert, and A. Bruhn. From box filtering to fast explicit diffusion. In M. Goesele, S. Roth, A. Kuijper, B. Schiele, and K. Schindler, editors, *Pattern Recognition*, volume 6376 of *Lecture Notes in Computer Science*, pages 533–542. Springer, Berlin, 2010.

[88] H. Grossauer and O. Scherzer. Using the complex Ginzburg–Landau equation for digital impainting in 2D and 3D. In L. D. Griffin and M. Lillholm, editors, *Scale-Space Methods in Computer Vision*, volume 2695 of *Lecture Notes in Computer Science*, pages 225–236. Springer, Berlin, 2003.

[89] S. Heinzel. Image compression with osmosis. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[90] N. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, second edition, 2002.

[91] L. Hoeltgen. *Optimal interpolation data for image reconstructions*. PhD thesis,

Department of Computer Science, Saarland University, Saarbrücken, Germany, 2014.

[92] L. Hoeltgen, S. Setzer, and J. Weickert. An optimal control approach to find sparse data for Laplace interpolation. In A. Heyden, F. Kahl, C. Olsson, M. Oskarsson, and X.-C. Tai, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 8081 of *Lecture Notes in Computer Science*, pages 151–164. Springer, Berlin, 2013.

[93] L. Hoeltgen and J. Weickert. Why does non-binary mask optimisation work for diffusion-based image compression? In X.-C. Tai, E. Bae, T. F. Chan, S. Y. Leung, and M. Lysaker, editors, *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, volume 8932 of *Lecture Notes in Computer Science*, pages 85–98. Springer, Berlin, 2015.

[94] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[95] Paul G Howard, Faouzi Kossentini, Bo Martins, Søren Forchhammer, and William J Rucklidge. The emerging JBIG2 standard. *IEEE Transactions on Circuits, Systems and Video Technology*, 8(7):838–848, November 1998.

[96] D. A. Huffman. A method for the construction of minimum redundancy codes. *Proceedings of the IRE*, 40:1098–1101, 1952.

[97] R. Hummel and R. Moniot. Reconstructions from zero-crossings in scale space. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:2111–2130, 1989.

[98] F. Huth. Shape coding with quadrupoles. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2011.

[99] F. Jager. Contour-based segmentation and coding for depth map compression. In *Visual Communications and Image Processing*, pages 1–4, Tainan City, Taiwan, November 2011.

[100] P. Johansen, S. Skelboe, K. Grue, and J. D. Andersen. Representing signals by their toppoints in scale space. In *Proc. Eighth International Conference on Pattern Recognition*, pages 215–217, Paris, France, October 1986.

[101] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Studentlitteratur, Lund, 1987.

[102] Joint Bi-level Image Experts Group. Information technology – progressive lossy/lossless coding of bi-level images. ISO/IEC JTC1 11544, ITU-T Rec. T.82, 1993. Final Committee Draft 11544.

[103] S. H. Kang and R. March. Variational models for image colorization via chromaticity and brightness decomposition. *IEEE Transactions on Image Processing*, 16(9):2251–2261, September 2007.

[104] F. M. W. Kanters, M. Lillholm, R. Duits, B. J. P. Jansen, B. Platel, L.M.J. Florack, and B. M. ter Haar Romeny. On image reconstruction from multiscale top points. In R. Kimmel, N. Sochen, and J. Weickert, editors, *Scale Space and*

PDE Methods in Computer Vision, volume 3459 of *Lecture Notes in Computer Science*, pages 431–439. Springer, Berlin, 2005.

[105] L. Kaufhold. Variational methods for the colourisation of greyscale images. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[106] N. Kaulgud and U.B. Desai. Image zooming: Use of wavelets. In S. Chaudhuri, editor, *Super-Resolution Imaging*, volume 632 of *The International Series in Engineering and Computer Science*, pages 21–44. Springer, New York, 2002.

[107] R. Kazinnik, S. Dekel, and N. Dyn. Low bit-rate image coding using adaptive geometric piecewise polynomial approximation. *IEEE Transactions on Image Processing*, 16(9):2225–2233, 2007.

[108] J. Kohout. On digital image representation by the Delaunay triangulation. In D. Mery and L. Rueda, editors, *Advances in Image and Video Technology*, volume 4872 of *Lecture Notes in Computer Science*, pages 826–840. Springer, Berlin, 2007.

[109] I. Kopilovic and T. Szirányi. Artifact reduction with diffusion preprocessing for image compression. *Optical Engineering*, 44(2):1–14, February 2005.

[110] H. Köstler, M. Stürmer, C. Freundl, and U. Rüde. PDE based video compression in real time. Technical Report 07–11, Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich–Alexander–Universität Erlangen–Nürnberg, Germany, 2007.

[111] M. Kunt, A. Ikonomopoulos, and M. Kocher. Second-generation image-coding techniques. *Proceedings of the IEEE*, 73(4):549–574, April 1985.

[112] M. Kunt and O. Johnsen. Block coding of graphics: A tutorial review. *Proceedings of the IEEE*, 68(7):770–786, July 1980.

[113] B. Kurt, M. Gökmen, and A. K. Jain. Image compression based on Centripede Model. In A. Del Bimbo, editor, *Image Analysis and Processing*, volume 1310 of *Lecture Notes in Computer Science*, pages 303–310. Springer, Berlin, 1997.

[114] T. Lehmann, C. Gönner, and K. Spitzer. Survey: Interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, 18(11):1049–1075, November 1999.

[115] B. Lehner, G. Umlauf, and B. Hamann. Image compression using data-dependent triangulations. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, N. Paragios, S.-M. Tanveer, T. Ju, Z. Liu, S. Coquillart, C. Cruz-Neira, T. Müller, and Tom Malzbender, editors, *Advances in Visual Computing*, volume 4841 of *Lecture Notes in Computer Science*, pages 351–362. Springer, Berlin, 2007.

[116] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *Proc. SIGGRAPH 2004*, pages 689–694, New York, NY, August 2004.

[117] X. Li. Anisotropic mesh adaptation for image representation and scaling. arXiv:1402.4893v1 [cs.CV], February 2014.

[118] Y. Li, M. Sjostrom, U. Jennehag, and R. Olsson. A scalable coding approach for high quality depth image compression. In *Proc. 3DTV-Conference: The True*

Vision – Capture, Transmission and Display of 3D Video, Zurich, Switzerland, October 2012. IEEE.

[119] Y. Li, M. Sjostrom, U. Jennehag, and R. Olsson. A scalable coding approach for high quality depth image compression. In *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 1 –4, October 2012.

[120] M. Lillholm, M. Nielsen, and L. D. Griffin. Feature-based image analysis. *International Journal of Computer Vision*, 52(2/3):73–95, 2003.

[121] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang. Image compression with edge-based inpainting. *IEEE Transactions on Circuits, Systems and Video Technology*, 17(10):1273–1286, October 2007.

[122] R. Lund. 3-D data compression with homogeneous diffusion. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2011.

[123] M. Mahoney. Adaptive weighing of context models for lossless data compression. Technical Report CS-2005-16, Florida Institute of Technology, Melbourne, FL, December 2005.

[124] M. Mainberger. *PDE-based Image Compression Based on Edges and Optimal Data*. PhD thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2014.

[125] M. Mainberger, A. Bruhn, J. Weickert, and S. Forchhammer. Edge-based image compression of cartoon-like images with homogeneous diffusion. *Pattern Recognition*, 44(9):1859–1873, September 2011.

[126] M. Mainberger, C. Schmaltz, M. Berg, J. Weickert, and M. Backes. Diffusion-based image compression in steganography. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, C. Fowlkes, S. Wang, M.-H. Choi, S. Mantler, J. Schulze, D. Acevedo, K. Mueller, and M. Papka, editors, *Advances in Visual Computing*, volume 7432 of *Lecture Notes in Computer Science*, pages 219–228. Springer, Berlin, July 2012.

[127] F. Malgouyres and F. Guichard. Edge direction preserving image zooming: A mathematical and numerical analysis. *SIAM Journal on Numerical Analysis*, 39(1):1–37, 2001.

[128] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, San Diego, second edition, 1999.

[129] S.G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12), December 1993.

[130] R. March. Computation of stereo disparity using regularization. *Pattern Recognition Letters*, 8:181–187, October 1988.

[131] S. Masnou and J.-M. Morel. Level lines based disocclusion. In *Proc. 1998 IEEE International Conference on Image Processing*, volume 3, pages 259–263, Chicago, IL, October 1998.

[132] J. Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, March 1960.

[133] E. Meijering. A chronology of interpolation: From ancient astronomy to modern signal and image processing. *Proceedings of the IEEE*, 90(3):319–342, March 2002.

[134] A. Meister. *Numerik linearer Gleichungssysteme: eine Einführung in moderne Verfahren.* Vieweg, 2008.

[135] Y.A. Melnikov and M.Y. Melnikov. *Green's Functions: Construction and Applications.* De Gruyter, Berlin, 2012.

[136] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[137] J. Modersitzki. *Numerical Methods for Image Registration.* Oxford University Press, Oxford, 2004.

[138] M. Moinard, I. Amonou, P. Brault, and P. Duhamel. Image and video compression scheme based on the prediction of transformed coefficients. In *Proc. Seventh International Symposium on Image and Signal Processing and Analysis*, pages 385–389, Dubrovnik, Croatia, September 2011.

[139] K. W. Morton and L. M. Mayers. *Numerical Solution of Partial Differential Equations.* Cambridge University Press, Cambridge, UK, 1994.

[140] Y. Morvan, P. H. N. de With, and D. Farin. Platelet-based coding of depth maps for the transmission of multiview images. In A. J. Woods, N. A. Dodgson, J. O. Merritt, M. T. Bolas, and I. E. McDowall, editors, *Proceedings of SPIE: Stereoscopic Displays and Applications*, volume 6055, San Jose, California, USA, January 2006.

[141] P. Mrázek. *Nonlinear Diffusion for Image Filtering and Monotonicity Enhancement.* PhD thesis, Czech Technical University, Prague, Czech Republic, June 2001.

[142] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:565–593, 1986.

[143] F. Nedwed. A probabilistic approach to image compression using subdivisions. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2015.

[144] Y. Nesterov and I.U.E. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course.* Applied Optimization. Springer, Berlin, 2004.

[145] T. Nguyen and D. Marpe. Performance analysis of HEVC-based intra coding for still image compression. In M. Domanski, T. Grajek, D. Karwowski, and R. Stasinski, editors, *Picture Coding Symposium*, pages 233–236, Kraków, Poland, May 2012.

[146] G. M. Nielson and J. Tvedt. Comparing methods of interpolation for scattered volumetric data. In D. F. Rogers and R. A. Earnshaw, editors, *State of the Art in Computer Graphics: Aspects of Visualization*, pages 67–86. Springer, New York, 1994.

[147] J. Niu. A steganography inspired application of hiding a high resolution image in its low resolution version. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2015.

[148] P. Ochs, Y. Chen, T. Brox, and T. Pock. iPiano: Inertial proximal algorithm for non-convex optimization. *SIAM Journal on Imaging Sciences*, 7(2):1388–1419, 2014.

[149] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, March 1982.

[150] W. B. Pennebaker and J. L. Mitchell. *JPEG: Still Image Data Compression Standard*. Springer, New York, 1992.

[151] P. Peter. Three-dimensional data compression with anisotropic diffusion. In J. Weickert, M. Hein, and B. Schiele, editors, *Pattern Recognition*, volume 8142 of *Lecture Notes in Computer Science*, pages 231–236. Springer, Berlin, 2013.

[152] P. Peter, L. Kaufhold, and J. Weickert. Turning diffusion-based image colourisation into efficient colour compression. Technical Report 370, Department of Mathematics, Saarland University, Saarbrücken, Germany, December 2015.

[153] P. Peter, C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert. Beyond pure quality: Progressive mode, region of interest coding and real time video decoding in PDE-based image compression. Technical Report 354, Department of Mathematics, Saarland University, Saarbrücken, Germany, January 2015.

[154] P. Peter and J. Weickert. Colour image compression with anisotropic diffusion. In *Proc. 21st IEEE International Conference on Image Processing*, pages 4822–4826, Paris, France, October 2014.

[155] P. Peter and J. Weickert. Compressing images with diffusion- and exemplar-based inpainting. In J.-F. Aujol, M. Nikolova, and N. Papadakis, editors, *Scale Space and Variational Methods in Computer Vision*, volume 9087 of *Lecture Notes in Computer Science*, pages 154–165. Springer, Berlin, 2015.

[156] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1762–1769, 2011.

[157] M. Puhl. Edge based image compression with mixed boundary conditions. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2012.

[158] S. D. Rane, G. Sapiro, and M. Bertalmío. Structure and texture filling-in of missing image blocks in wireless transmission and compression applications. *IEEE Transactions on Image Processing*, 12(3):296–302, March 2003.

[159] K. R. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[160] J. J. Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, 20(3):198–203, May 1976.

[161] J. J. Rissanen and G. G. Langdon Jr. Arithmetic coding. *IBM Journal of Research and Development*, 23(2):149–162, March 1979.

[162] A. Roussos and P. Maragos. Vector-valued image interpolation by an anisotropic diffusion-projection PDE. In F. Sgallari, F. Murli, and N. Paragios, editors, *Scale Space and Variational Methods in Computer Vision*, volume 4485 of *Lecture Notes in Computer Science*, pages 104–115. Springer, Berlin, 2007.

[163] J. Ruiz-Hidalgo, J. R. Morros, P. Aflaki, F. Calderero, and F. Marqués. Multiview depth coding based on combined color/depth segmentation. *Journal of Visual Communication and Image Representation*, 23(1):42–52, January 2012.

[164] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Thomson Learning, London, 1996.

[165] T. Sauer. *Numerical Analysis.* Pearson Addison Wesley, Boston, 2006.

[166] A. Scheer. Entropy coding for PDE-based image compression. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2014.

[167] O. Scherzer and J. Weickert. Relations between regularization and diffusion filtering. *Journal of Mathematical Imaging and Vision*, 12(1):43–63, February 2000.

[168] C. Schmaltz, P. Gwosdek, A. Bruhn, and J. Weickert. Electrostatic halftoning. *Computer Graphics Forum*, 29(8):2313–2327, December 2010.

[169] C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert. Progressive modes in PDE-based image compression. In *Proc. 30th Picture Coding Symposium*, pages 233–236, San Jose, CA, December 2013. IEEE.

[170] C. Schmaltz, P. Peter, M. Mainberger, F. Ebel, J. Weickert, and A. Bruhn. Understanding, optimising, and extending data compression with anisotropic diffusion. *International Journal of Computer Vision*, 108(3):222–240, July 2014.

[171] C. Schmaltz and J. Weickert. Video compression with 3-D pose tracking, PDE-based image coding, and electrostatic halftoning. In A. Pinz, T. Pock, H. Bischof, and F. Leberl, editors, *Pattern Recognition*, volume 7476 of *Lecture Notes in Computer Science*, pages 438–447. Springer, Berlin, 2012.

[172] C. Schmaltz, J. Weickert, and A. Bruhn. Beating the quality of JPEG 2000 with anisotropic diffusion. In J. Denzler, G. Notni, and H. Süße, editors, *Pattern Recognition*, volume 5748 of *Lecture Notes in Computer Science*, pages 452–461. Springer, Berlin, 2009.

[173] M. Schneider. Image compression with homogeneous diffusion inpainting on level lines. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[174] M. Schneider. Image compression with sparse gradient data. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2015.

[175] C. Schroers, S. Setzer, and J. Weickert. A variational taxonomy for surface

reconstruction from oriented points. *Computer Graphics Forum*, 33(5):195–204, August 2014.

[176] M. Schwinn. Patch-based inpainting and compression. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[177] S. Setzer, G. Steidl, and J. Morgenthaler. On cyclic gradient descent reprojection. *Computational Optimization and Applications*, 54(2):417–440, March 2013.

[178] J. Shen and T. F. Chan. Mathematical models for local nontexture inpaintings. *SIAM Journal on Applied Mathematics*, 62(3):1019–1043, 2002.

[179] B. C. Smith. Eigenvalues and eigenvectors of the discrete Laplacian. `http://math.ucdenver.edu/~brysmith/software/Eigenvalues_of_the_discrete_laplacian_bryan_smith.pdf`, 2012. Last visited September 02, 2015.

[180] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing.* California Technical Publishing, San Diego, CA, USA, 1997.

[181] A. Solé, V. Caselles, G. Sapiro, and F. Arandiga. Morse description and geometric encoding of digital elevation maps. *IEEE Transactions on Image Processing*, 13(9):1245–1262, September 2004.

[182] P. Solin and D. Andrs. On scientific data and image compression based on adaptive higher-order FEM. *Advances in Applied Mathematics and Mechanics*, 1(1):56–68, February 2009.

[183] J. Steil. Shape coding with nonapoles. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[184] G. Strang. *Introduction to Applied Mathematics.* Wellesley-Cambridge Press, Wellesley, 1986.

[185] G. Strang. The discrete cosine transform. *SIAM Review*, 41(1):135–147, 1999.

[186] G. Strang and S. MacNamara. Functions of difference matrices are Toeplitz plus Hankel. *SIAM Review*, 56(3):525–546, 2014.

[187] P. Strobach. Quadtree-structured recursive plane decomposition coding of images. *IEEE Transactions on Signal Processing*, 39(6):1380–1397, June 1991.

[188] G. J. Sullivan and R. J. Baker. Efficient quadtree coding of images and video. *IEEE Transactions on Image Processing*, 3(3):327–331, May 1994.

[189] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, December 2012.

[190] T. Szirányi, I. Kopilovic, and B. P. Tóth. Anisotropic diffusion as a preprocessing step for efficient image compression. In *Proc. 14th International Conference on Pattern Recognition*, volume 2, pages 1565–1567, Brisbane, Australia, August 1998. IEEE Computer Society Press.

[191] D. S. Taubman and M. W. Marcellin, editors. *JPEG 2000: Image Compression Fundamentals, Standards and Practice.* Kluwer, Boston, 2002.

[192] V. N. Temlyakov. Nonlinear methods of approximation. *Foundations of Computational Mathematics*, 3(1):33–107, 2003.

[193] J.W. Thomas. *Numerical Partial Differential Equations: Finite Difference Methods*. Graduate Texts in Mathematics. Springer, Berlin, 1995.

[194] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, June 1997.

[195] J.A. Tropp and A.C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 53(12):4655–4666, December 2007.

[196] D. Tschumperlé and R. Deriche. Vector-valued image regularization with PDEs: A common framework for different applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):506–516, April 2005.

[197] H. Tsuji, T. Sakatani, Y. Yashima, and N. Kobayashi. A nonlinear spatio-temporal diffusion and its application to prefiltering in MPEG-4 video coding. In *Proc. 2002 IEEE International Conference on Image Processing*, volume 1, pages 85–88, Rochester, NY, September 2002.

[198] K. Uhlir and V. Skala. Reconstruction of damaged images using radial basis functions. In *Proc. 13th European Signal Processing Conference (EUSIPCO)*, pages 160–163, Antalya, Turkey, September 2005.

[199] M. Vetterli and A. Ligtenberg. A discrete Fourier-Cosine transform chip. *IEEE Journal on Selected Areas in Communications*, 4(1):49–61, September 2006.

[200] G. Voronoi. Nouvelles applications des paramétres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik*, 134:198–287, 1908.

[201] B. Šarler. Solution of potential flow problems by the modified method of fundamental solutions: Formulations with the single layer and the double layer fundamental solutions. *Engineering Analysis with Boundary Elements*, 33(12):1374–1382, 2009. Special Issue on the Method of Fundamental Solutions in honour of Professor Michael Golberg.

[202] C. Wang, X. Sun, F. Wu, and H. Xiong. Image compression with structure-aware inpainting. In *Proc. 2006 IEEE International Symposium on Circuits and Systems*, pages 1816–1819, Kos, Greece, May 2006.

[203] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[204] J. Weickert. Theoretical foundations of anisotropic diffusion in image processing. *Computing Supplement*, 11:221–236, 1996.

[205] J. Weickert. Coherence-enhancing diffusion of colour images. In A. Sanfeliu, J. J. Villanueva, and J. Vitrià, editors, *Proc. Seventh National Symposium on Pattern Recognition and Image Analysis*, volume 1, pages 239–244, Barcelona, Spain, April 1997.

[206] J. Weickert. *Anisotropic Diffusion in Image Processing.* Teubner, Stuttgart, 1998.

[207] J. Weickert. Coherence-enhancing diffusion filtering. *International Journal of Computer Vision*, 31(2/3):111–127, April 1999.

[208] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn. Cyclic schemes for PDE-based image analysis. Technical Report 327, Department of Mathematics, Saarland University, Saarbrücken, Germany, April 2015.

[209] J. Weickert and M. Welk. Tensor field interpolation with PDEs. In J. Weickert and H. Hagen, editors, *Visualization and Processing of Tensor Fields*, pages 315–325. Springer, Berlin, 2006.

[210] H. Werner. Studies on contour: I. qualitative analyses. *The American Journal of Psychology*, 47(1):40–64, January 1935.

[211] Y. Wu, H. Zhang, Y. Sun, and H. Guo. Two image compression schemes based on image inpainting. In *Proc. 2009 International Joint Conference on Computational Sciences and Optimization*, pages 816–820, Sanya, China, April 2009.

[212] Z. Xie, W. R. Franklin, B. Cutler, M. A. Andrade, M. Inanc, and D. M. Tracy. Surface compression using over-determined Laplacian approximation. In F. T. Luk, editor, *Advanced Signal Processing Algorithms, Architectures, and Implementations XVII*, volume 6697 of *Proceedings of SPIE.* SPIE Press, Bellingham, 2007.

[213] Z. W. Xiong, X. Y. Sun, F. Wu, and S. P. Li. Image coding with parameter-assistant inpainting. In *Proc. 2007 IEEE International Conference on Image Processing*, volume 2, pages 369–372, San Antonio, TX, September 2007.

[214] S. Yang and Y.-H. Hu. Coding artifact removal using biased anisotropic diffusion. In *Proc. 1997 IEEE International Conference on Image Processing*, volume 2, pages 346–349, Santa Barbara, CA, October 1997.

[215] S. Yao, W. Lin, Z. Lu, E. P. Ong, and X. Yang. Adaptive nonlinear diffusion processes for ringing artifacts removal on JPEG 2000 images. In *Proc. 2004 IEEE International Conference on Multimedia and Expo*, pages 691–694, Taipei, Taiwan, June 2004.

[216] V. Yatnalli and K. L. Sudha. Image compression with inpainting. In *Proc. IEEE International Symposium on Signal Processing and Information Technology*, pages 158–163, Ho Chi Minh City, Vietnam, December 2012.

[217] N. Yokoya. Surface reconstruction directly from binocular stereo images by multiscale-multistage regularization. In *Proc. Eleventh International Conference on Pattern Recognition*, volume 1, pages 642–646, The Hague, The Netherlands, August 1992.

[218] P. Zanuttigh and G. M. Cortelazzo. Compression of depth information for 3D rendering. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 1–4, Potsdam, Germany, May 2009.

[219] Y. Zeevi and D. Rotem. Image reconstruction from zero-crossings. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34:1269–1277, 1986.

[220] G. Zeng and N. Ahmed. A block coding technique for encoding sparse binary patterns. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(5):778–780, May 1989.

[221] C. Zhao and M. Du. Image compression based on PDEs. In *Proc. 2011 International Conference of Computer Science and Network Technology*, pages 1768–1771, Harbin, China, December 2011.

[222] H. Zimmer. PDE-based image compression using corner information. Master thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2007.

[223] S. Zimmer. Textures in PDE-based image compression. Bachelor thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013.

[224] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. A. J. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. In J. C. Hart, editor, *ACM Transactions on Graphics*, volume 23, pages 600–608, New York, USA, August 2004.