Saarland University

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

Understanding and Advancing PDE-based Image Compression

A Dissertation Submitted Towards the Degree Doctor of Engineering (Dr.-Ing.)
of the Faculties of Natural Sciences and Technology of Saarland University

submitted by

Pascal Peter

Saarbrücken, 2016

Day of Colloquium:

29.04.2016

Dean of Faculty:

Univ.-Prof. Dr. Frank-Olaf Schreyer

Chair of the Committee:

Prof. Dr. Antonio Krüger

Reviewers:

Dr. habil. Hanno Scharr
Prof. Joan Serra-Sagristà
Prof. Joachim Weickert

Academic Assistant:

Dr. Peter Ochs

Short Abstract

This thesis is dedicated to image compression with partial differential equations (PDEs). PDE-based codecs store only a small amount of image points and propagate their information into the unknown image areas during the decompression step. For certain classes of images, PDE-based compression can already outperform the current quasi-standard, JPEG2000. However, the reasons for this success are not yet fully understood, and PDE-based compression is still in a proof-of-concept stage. With a probabilistic justification for anisotropic diffusion, we contribute to a deeper insight into design principles for PDE-based codecs. Moreover, by analysing the interaction between efficient storage methods and image reconstruction with diffusion, we can rank PDEs according to their practical value in compression. Based on these observations, we advance PDE-based compression towards practical viability: First, we present a new hybrid codec that combines PDE- and patch-based interpolation to deal with highly textured images. Furthermore, a new video player demonstrates the real-time capacities of PDE-based image interpolation and a new region of interest coding algorithm represents important image areas with high accuracy. Finally, we propose a new framework for diffusion-based image colourisation that we use to build an efficient codec for colour images. Experiments on real world image databases show that our new method is qualitatively competitive to current state-of-the-art codecs.

Kurzzusammenfassung

Diese Dissertation ist der Bildkompression mit partiellen Differentialgleichungen (PDEs, partial differential equations) gewidmet. PDE-Codecs speichern nur einen geringen Anteil aller Bildpunkte und transportieren deren Information in fehlende Bildregionen. In einigen Fällen kann PDE-basierte Kompression den aktuellen Quasi-Standard, JPEG2000, bereits schlagen. Allerdings sind die Gründe für diesen Erfolg noch nicht vollständig erforscht, und PDE-basierte Kompression befindet sich derzeit noch im Anfangsstadium. Wir tragen durch eine probabilistische Rechtfertigung anisotroper Diffusion zu einem tieferen Verständnis PDE-basierten Codec-Designs bei. Eine Analyse der Interaktion zwischen effizienten Speicherverfahren und Bildrekonstruktion erlaubt es uns, PDEs nach ihrem Nutzen für die Kompression zu beurteilen. Anhand dieser Einsichten entwickeln wir PDE-basierte Kompression hinsichtlich ihrer praktischen Nutzbarkeit weiter: Wir stellen einen Hybrid-Codec für hochtexturierte Bilder vor, der umgebungsbasierte Interpolation mit PDEs kombiniert. Ein neuer Video-Dekodierer demonstriert die Echtzeitfähigkeit PDE-basierter Interpolation und eine Region-of-Interest-Methode erlaubt es, wichtige Bildbereiche mit hoher Genauigkeit zu speichern. Schlussendlich stellen wir ein neues diffusionsbasiertes Kolorierungsverfahren vor, welches uns effiziente Kompression von Farbbildern ermöglicht. Experimente auf Realwelt-Bilddatenbanken zeigen die Konkurrenzfähigkeit dieses Verfahrens auf.

Abstract

The omnipresence of digital images and movies on computers and entertainment devices, particularly on the internet, implies a necessity for efficient storage of this kind of data. Today, image and video coding is dominated by transform-based methods like JPEG [215], JPEG2000 [277] and the MPEG family [143]. These methods rely on sparse representations by means of a discrete cosine or wavelet transform. Recently, a completely different approach has challenged these established encoders: Compression with PDEs (partial differential equations) creates sparsity directly in the spatial domain by storing only a few selected pixels. For decompression, the codec reconstructs the missing image parts by propagating information by a physics-inspired diffusion process. On specific classes of images, PDE-based compression can already surpass the quality of JPEG and JPEG2000 [250]. This thesis aims to provide a deeper understanding of this success, assess which tasks remain to be solved, and pave the way for practical viability of PDE-based compression.

Currently, the state of the art in PDE-based compression relies on edge-enhancing anisotropic diffusion (EED) [295] which has exceptional interpolation capabilities. In order to get a better understanding of this codec's success, we justify the need for PDEs that allow directionally-dependent propagation from the statistics of natural images. By relating diffusion to Bayesian approaches for image processing, we construct a unifying framework for eight existing diffusion models. We show that the historical evolution of diffusion models reflects an increasingly accurate approximation of image statistics by the corresponding Bayesian priors. Our experiments for denoising and inpainting on a large database of natural images underline these results.

Apart from PDE-based reconstruction, many other compression steps are vital for a successful codec: The selection of known data, entropy coding, and quantisation have a large effect on the overall coding performance. To foster understanding for proper design of PDE-based codecs, we carefully analyse the interplay between different compression components and PDEs. To this end, we propose two new codecs that each rely on a different core strategy. Our first codec selects pixel-accurate known data that allows accurate reconstructions, but is expensive to store. The second approach restricts the choice to a regular adaptive grid which can be efficiently stored as a binary tree at the expense of suboptimal known data. Our evaluations with the most popular PDEs in compression (harmonic, biharmonic, and edge-enhancing diffusion) yield a surprising result: The inpainting capabilities of these PDEs do not always determine their success in a given codec, if other compression steps like quantisation are involved. In particular, harmonic diffusion is sensitive to the location of known data, while the quality of biharmonic inpainting suffers under coarse quantisation. In contrast, EED is robust with respect to both.

Our investigations of key components for PDE-based compression allow us to address several missing features and limitations of the current generation of diffusion coders. First, we focus on the fact that interpolation with PDEs inherently relies on a smoothness assumption

on the missing image parts. Since fine-scale texture violates this assumption, it is difficult to compress. As a remedy, we propose a block-decomposition method that combines PDE-based reconstruction with patch-based approach for texture synthesis. This hybrid method can improve performance on highly textured images significantly and helps to close the gap between PDE-based codecs and JPEG2000 on this class of images.

Another current limitation of PDE-based compression is their runtime. In order to show that this class of codecs can also be used for time critical applications, we present a PDE-based video codec that allows real-time playback. We achieve this performance on consumer hardware by parallelisation, caching, and the reuse of already decompressed frames as an initialisation for upcoming frames. We show that the video quality does not deteriorate visibly in comparison to decoding without time constraints on the entire movie *Nosferatu* [87].

Finally, we address perceptive coding. We introduce a region of interest mode to PDE-based compression. A user or an automated algorithm can specify a pixel-accurate, continuous weighting of different regions in the compressed image. According to this weighting the error is distributed throughout the image. We demonstrate the applications with examples from medical imaging, photography and video encoding. An even more important aspect of perceptive coding is the higher sensitivity of the human visual system w.r.t. structural information in images than w.r.t. colour. This fact is already exploited by JPEG and JPEG2000, but colour images have not been addressed with a dedicated codec in the proof-of-concept stage of PDE-based compression. As a remedy, we introduce a new framework for diffusion-based colourisation: A user manually specifies a few colour strokes in a grey value image, and diffusion fills in the missing colours. To this end, we use a YCbCr colour space to decompose the image into luma (brightness) and chroma (colour) channels. In order to exploit the correlation of these components, we guide the diffusion in the chroma channels by the structural information of the brightness component. Experiments with four different diffusion models show that our new anisotropic schemes outperform existing colourisation methods significantly. We combine this colourisation with our efficient coding techniques from the previous contributions into a luma preference codec. We dedicate a large amount of our bit budget to an accurate representation of the luma channel, which allows to reconstruct the chroma information with a small amount of known data. Due to performance increases inspired by our real-time video decoding, we are able to show for the first time that PDE-based compression is competitive to transform based coders on real world image databases.

In summary, we have fostered the understanding of PDE-based compression by a new probabilistic justification of anisotropic diffusion and a thorough analysis of the interaction between relevant codec components. This enables us to introduce new features and codecs that address textured images, real-time performance and perceptive coding.

Zusammenfassung

Die Allgegenwärtigkeit digitaler Bilder und Filme auf Computern und Unterhaltungselektronik legt die Notwendigkeit effizienter Speichermethoden nahe. Heutzutage wird Bild- und Videokompression von transformationsbasierten Methoden wie JPEG [215], JPEG2000 [277] und der MPEG-Familie [143] dominiert. In jüngerer Zeit forderte ein vollständig andersartiger Ansatz diese etablierten Methoden heraus: Kompression mit partiellen Differentialgleichungen (PDEs, partial differential equations) erzeugt durch die Speicherung einiger weniger, ausgewählter Bildpunkte eine kompakte Darstellung ohne Anwendung einer Transformation. Zur Dekompression werden die bekannten Informationen mittels eines physikalisch motivierten Diffusionsprozesses in die fehlenden Bildteile transportiert. PDE-basierte Kompression ist bereits nachweislich in der Lage, JPEG und JPEG2000 auf bestimmten Arten von Bildern zu übertreffen [250]. Diese Dissertation hat zum Ziel, diesen Erfolg besser zu verstehen, zu analysieren, welche Aufgaben noch zu bewältigen bleiben und den Weg für praxistaugliche Kompression mit PDEs zu ebnen.

Zur Zeit beruht der Stand der Kunst im Bereich der PDE-basierten Kompression auf kantenverstärkender anisotroper Diffusion (EED, edge-enhancing diffusion) [295], welche hervorragende Interpolationsfähigkeiten aufweist. Um ihren Erfolg besser zu verstehen, motivieren wir die Notwendigkeit richtungsabhängiger PDEs über die Statistik natürlicher Bilder. Indem wir eine Verbindung zwischen Bayes'schen Ansätzen der Bildverarbeitung und Diffusion herstellen, können wir acht existierenden Diffusionsmodelle in einen gemeinsamen theoretischen Rahmen einbetten. Wir zeigen, dass die historische Evolution von Diffusionsmodellen als eine zunehmend exaktere Annäherung an die Statistik der zugehörigen Bayes'schen A-Priori-Annahmen interpretiert werden kann. Unsere Experimente mit Bildentrauschung und Bildrekonstruktion auf einer umfangreichen Datenbank natürlicher Bilder unterstreichen diese Resultate.

Neben PDE-basierter Bilredkonstruktion ist eine Vielzahl anderer Kompressionsschritte von Bedeutung für den Erfolg eines Codecs. Sowohl die Wahl der bekannten Daten, als auch Entropie-Kodierung und Quantisierung beeinflussen die Leistung des Kodierers. Um PDE-basierte Codecs effizient gestalten zu können, analysieren wir das Zusammenspiel zwischen diesen Komponenten und verschiedenen PDEs. Wir führen zwei neue Codecs ein, die jeweils auf verschiedenen Strategien beruhen. Unser erstes Verfahren wählt bekannte Daten pixelgenau aus und erlaubt exakte Rekonstruktionen auf Kosten des Speicherplatzes. Der zweite Ansatz schränkt die Wahl auf ein reguläres adaptives Gitter ein. Diese suboptimalen Positionen können in Form eines Binärbaumes effizient gespeichert werden. Ein Vergleich häufig zur Kompression genutzten PDEs (harmonische, biharmonische und kantenverstärkende Diffusion) führt zu einem überraschenden Ergebnis: Die Interpolationsfähigkeiten dieser PDEs sind noch keine Garantie für effiziente Kompression. Harmonische Diffusion ist empfindlich gegenüber der Position bekannter Daten, wohingegen die Resultate biharmonischer Rekonstruktion unter grober Quantisierung leidet. Im Gegensatz dazu verhält sich EED in beiden Fällen robust.

Unsere Untersuchung der Kernkomponenten PDE-basierter Codecs erlaubt uns, mehrere Grenzen und fehlende Funktionen der aktuellen Generation von Diffusions-Kodierern in Angriff zu nehmen. Zunächst wenden wir uns dem Grundproblem texturierter Bilder zu. Da die Interpolation mit PDEs auf einer Glattheitsannahme beruht, können feinskalige Texturen nur schwer rekonstruiert werden. Um Abhilfe zu schaffen, stellen wir eine Blockdekompositionsmethode vor, die PDE-basierte Rekonstruktion mit umgebungsbasierter Textursynthese kombiniert. Dieser Hybrid-Ansatz erlaubt uns, auf hochtexturierten Bildern zu JPEG2000 aufzuschließen.

Ein weiterer begrenzender Faktor für PDE-basierte Kompression ist ihre Laufzeit. Um zu zeigen, dass diese Klasse von Kodierern auch für zeitkritische Anwendungen eingesetzt werden kann, führen wir einen neuen PDE-basierten Video-Codec mit Echtzeitwiedergabe ein. Wir erreichen die hierzu nötige Geschwindigkeit auf handelsüblicher Hardware mittels Parallelisierung, Pufferung und Initialisierung mit bereits dekomprimierten Einzelbildern. Anhand des Filmes *Nosferatu* [87] zeigen wir, dass die Qualität nicht sichtbar unter der Zeitrestriktion leidet.

Schlussendlich widmen wir uns wahrnehmungsbasierter Kompression. Zunächst führen wir einen „Region-of-Interest“-Modus für PDE-basierte Kodierer ein. Er ermöglicht Nutzern oder automatisierten Algorithmen, den Rekonstruktionsfehler verschiedener Bildareale zu gewichten. Wir demonstrieren verschiedene Anwendungsfälle der medizinischen Bildgebung, Fotografie und Video-Kompression. Zudem nutzen wir aus, das Struktur für die menschliche Wahrnehmung wichtiger ist als Farbe. JPEG und JPEG2000 nutzen dieses Phänomen bereits, doch PDE-basierte Kompression verfügt bisher noch nicht über einen dedizierten Farbmodus. Um Abhilfe zu schaffen, widmen wir uns zunächst der Bildkolorierung: Ein Nutzer gibt einige wenige Farbstriche in einem Grauwertbild vor, und Diffusion füllt den Rest des Bildes mit Farbe. Mittels eines YCbCr-Farbraumes unterteilen wir das Bild zunächst in Luma-(Helligkeits-) und Chroma-(Farb-)Kanäle. Um Korrelationen zwischen diesen Komponenten auszunutzen, steuern wir die Diffusion in den Chromakanälen anhand der Struktur der Helligkeitskomponente. Experimente mit vier verschiedenen Diffusionsmodellen zeigen, dass unsere neuen anisotropen Verfahren existierende Kolorierungsmodelle erheblich übertreffen. Wir kombinieren diese Kolorierung mit effizienten Speichermethoden zu einem Luma-Vorzugs-Codec. Die Luma-Information wird mit einem umfangreichen Speicherbudget exakt repräsentiert, wodurch die Chroma-Informationen aus einer geringen Menge von Farbinformationen wiederhergestellt werden kann. Zum ersten Mal stellen wir die Konkurrenzfähigkeit PDE-basierte Kompression auf einer Realwelt-Bilddatenbank unter Beweis.

Zusammengefasst haben wir durch eine probabilistische Motivation anisotroper Diffusion und eine detaillierte Analyse der Interaktion verschiedener relevanter Kernkomponenten erfolgreicher Kodierer ein tieferes Verständnis für PDE-basierte Kompression erreicht. Dies ermöglicht es uns, neue Funktionen und Kompressionsverfahren zu entwickeln, die für texturierte Bilder, Echtzeitanwendungen und wahrnehmungsbasierte Kompression geeignet sind.

Acknowledgements

> *Words are easy, like the wind;*
> *Faithful friends are hard to find.*
>
> — **William Shakespeare**
> (Dramatist)

First and foremost I want to thank my advisor *Prof. Joachim Weickert*. Not only did he always provide guidance when needed, he has also offered me a unique working environment in his Mathematical Image Analysis (MIA) Group. I value very highly that I had a lot of freedom in research and great opportunities for teaching during my time as a Ph.D student.

Next, I want to thank my close collaborators and co-authors who have contributed directly to the scientific content of this thesis. In particular, I owe a lot to the mentorship of *Dr. Christian Schmaltz*. He has passed a lot of his experience in PDE-based compression, debugging, and dry humour on to me and it was always a pleasure to collaborate with him. *Dr. Laurent Hoeltgen* was always open for discussion, providing a broad mathematical knowledge, and generally being a pleasant company during roughly three years of sharing an office with me. As a fellow researcher in PDE-based image compression, *Sebastian Hoffmann* has not only collaborated directly with me, but was also an important dialogue partner with right to new ideas and challenges in compression. I also want to thank *Housen Li*, *Prof. Tatyana Krivobokova*, and *Prof. Axel Munk* for their confirmation and support in statistical matters. *Lilli Kaufhold* provided the foundations for diffusion-based colourisation and colour compression methods with her Bachelor's thesis. Moreover, *Frank Nedwed* kindly agreed to collaborate on expanding and publishing stochastic methods for image compression.

In addition to these direct contributors, there are many people who had an indirect impact on the thesis. First I want to thank *Sarah Diehl, Sarah Schäffer, Martin Schmidt, and Joachim Weickert* for proofreading my thesis and providing a lot of valuable feedback.

My gratitude for fruitful discussions and support also go to the current and former research staff of the MIA group who have not been mentioned above: *Dr. Matthias Augustin, Leif Bergerhoff, Prof. Michael Breuss, Prof. Andrés Bruhn, Marcelo Cárdenas, Dr. Oliver Demetz, David Hafner, Kai Hagenburg, Yong Chul Ju, Dr. Markus Mainberger, Sabine Müller, Dr. Peter Ochs, Markus Schneider, Cristopher Schroers, Dr. Simon Setzer, and a.o. Prof. Martin Welk*.

Furthermore, I want to thank the secretary of the MIA group, *Ellen Wintringer*, for keeping most of the tedious bureaucracy away from me, thus allowing me to invest more time into research. In a similar way, our current and former system administrators *Marcus Hargarter, Heinz-Ulrich Griehl, and Peter Franke* have saved me a lot of time dealing with technical troubles.

Last but not least, I want show my gratitude to my family. I thank *Birgit and Engelbert Peter*, as well as *Maria and Erwin Hoffmann* for their continuous and unconditional support during all of my studies. Finally, I am thankful for the patience that my fiancée *Sarah Diehl* had with me during the final stages of my dissertation and all the other big and small things she has done for me.

Contents

Introduction

1

> *Everything starts somewhere,*
> *although many physicists disagree.*
>
> — **Sir Terry Pratchett**
> (Writer)

Today, visual content is everywhere: Photographs and videos are shared over the internet, the entertainment industry produces movies with ever increasing resolution, and professional applications like medical imaging generate large amounts of 3-D volumetric data. Digitalisation is a common trend for all of these different sectors that produce visual content. Even the image acquisition process, which has remained an analogue procedure for many years after digital images became popular, is changing rapidly. The availability of analogue cameras in German households declined from 60% to 34,5% in the timespan from 2009 to 2014. In contrast, digital cameras are available in 75,6% of households in 2014. Moreover, most of the current mobile phones contain digital cameras. More than 90% of households are thereby equipped with the means to produce digital images [264].

Traditionally, the aforementioned transition from analogue to digital was a justification for research activities in compression (see e.g. [268]). Expensive storage media implied a need for efficient representations of digital images. Furthermore, moving data around on small portable media or over low bandwidth connections was cumbersome. Consider e.g. the capacity of a 1.44 MB floppy disk from 1987: Assuming that three byte are dedicated to the colour of every point in the image, an uncompressed 1024×768 colour image would have to be stored on two separate disks. However, with the rapid growth of hard disk space, the advent of cheap portable storage, and steadily increasing market penetration of broadband connections, this argument seems to have lost its relevance.

Nevertheless, there is still vivid research activity regarding compression. While storage and telecommunication technologies have been evolving quickly, there has also been a rapid development of image acquisition, home entertainment, and broadcasting technologies. Even compact smart phones can already produce images with $41 \cdot 10^6$ image points, which is roughly equivalent to 123 megabyte of uncompressed image data. In combination with the common use case of uploading image data directly from the smart phone to online services, it is clear that compression is still important. Other emerging technologies in image acquisition increase the amount of recorded data even further: Early lightfield cameras already have a resolution of 10^9 image points.

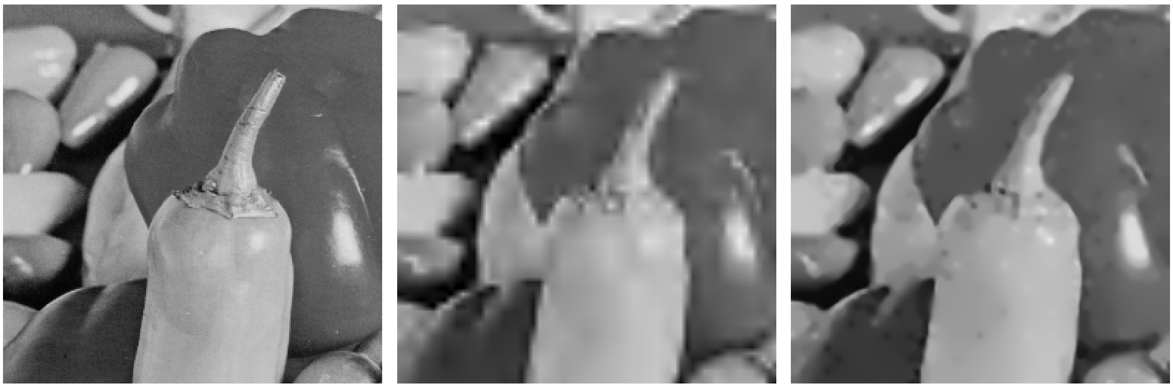
In video recording, this development is particularly extreme, since several factors increase simultaneously: The 4k standard increases the resolution of each video frame by a factor of four, and high frame rate video (HFR) almost triples the number of images per second. This combination of increasing fidelity of visual data and limited broadcast channels makes compression important. In traditional one-way broadcast with electromagnetic waves, frequency bands for digital television signals are reduced to free up other bands for mobile internet [160, p. 25]. Freeing up bandwidth while simultaneously maintaining the same perceived quality of digital TV is only one of the practical impacts of research in compression. Moreover, on-demand internet TV starts to replace traditional broadcasting. In 2013, video streaming was already responsible for 66% of the overall internet traffic and is projected to reach 79% by 2018 [61]. Counting traditional downloads and file sharing in addition, video data alone might even be responsible for 80-90% of the transmitted data.

Since the development and standardisation of the JPEG (Joint Photographic Experts Group) codec [215] in the 1980s, the compression community's predominant answer to the growing demand for efficient image encoding has been transform-based compression. This type of compression methods changes the representation of images. In its traditional representation, a digital image is a mapping of a finite number of spatial coordinates to colour or grey values. Instead, transform-based compression considers an image as a collection of coefficients that characterise image information on different scales. These coefficients have global or semi-local influence on the image and range from coarse image structures to highly variant detail. This allows for a flexible implementation of lossy compression, since details that are not important for the human visual system can be stored with reduced accuracy.

While JPEG is still widely used in the consumer market, successors like JPEG2000 (Joint Photographic Experts Group 2000) [277] and recently BPG (Better Portable Graphics) [88] are currently considered to be the state of the art in image compression. Since a large and active community of researchers was developing transform-based compression over many years and still continues to improve these quasi-standards, it is no easy task to challenge them.

However, during the last decade a fundamentally different approach for image compression that relies on partial differential equations (PDEs) has emerged. These PDE-based methods work directly with sparse representations in the spatial domain: They only store brightness or colour data at a few selected pixel locations. Carefully designed PDEs model the transport of this known information to the missing areas of the image, thus providing a way to reconstruct the full picture.

In 2005, Galić et al. [100] have proposed a proof of concept codec that could already surpass the quality of JPEG for specific imagery. Four years later, JPEG2000 was beaten by the rectangular subdivision with edge-enhancing diffusion (R-EED) codec of Schmaltz et al. [250]. Figure 1.1 illustrates that in particular for high compression ratios, PDE-based compression can achieve much more convincing results than its transform-based competitors. Both of these general-



original JPEG2000 R-EED

Figure 1.1.: **Comparison of R-EED and JPEG2000 on the grey value image *peppers*.** R-EED beats the quality of JPEG2000 by a considerable margin at a compression ratio of 80:1.

purpose encoders use edge-enhancing anisotropic diffusion (EED) of Weickert [295]. This PDE adapts the propagation of grey values to the local image structure. Other PDEs, such as the simple linear homogeneous diffusion model [140, 141], are particularly successful in niche-applications such as cartoon compression [184] or the compression of depth maps [127, 166]. However, such specialised codecs outperform the general purpose codec JPEG2000 even more significantly in their particular field as demonstrated by Figure 1.2.

While the aforementioned codecs are not the only relevant PDE-based compression methods, they provide a good impression of the quality that this class of methods can currently achieve. An in-depth review of the state of the art can be found in Section 3.3. The examples for qualitative comparisons between PDE-based and transform-based compression can convey the impression that codecs like R-EED have already reached a mature stage of development that puts them on equal footing with JPEG and JPEG2000 from a practical point of view. However, pure compression quality is not the only criterion that a successful codec has to fulfil.

A new competitor to JPEG and JPEG2000 might offer more accurate reconstructions, but does not necessarily provide all of the features that a user expects nowadays. For example, compression of colour data is a fundamental requirement for a successful compression method. However, the example in Figure 1.3 shows, that R-EED loses its advantage over JPEG2000 on colour data. If one considers images that contain a large amount of texture, R-EED can even fall behind JPEG (see Figure 1.4) on greyscale images. These examples show that for a fair comparison more categories than just reconstruction quality on a particular type of image must be considered.

In their detailed monograph on JPEG, Pennebaker and Mitchell [215, pp. 303–306] present a list of requirements that were the basis of the JPEG development and standardisation process. They also consider these goals as a good guideline for realistic comparisons of new codecs to JPEG:

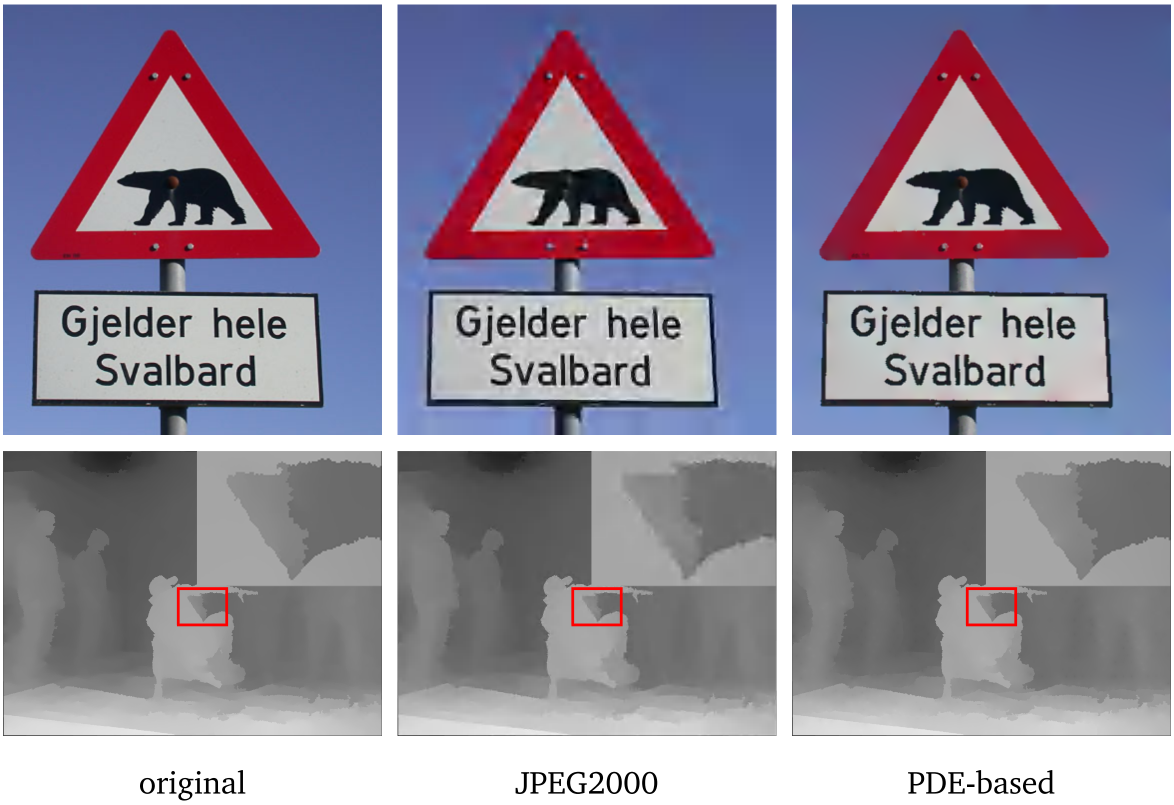


Figure 1.2.: **Cartoon and Depth Map Compression with PDEs.** The upper row shows a comparison of transform based coders and the cartoon compression codec by Mainberger et al. [184] at a compression ratio of 50:1. The bottom row contains a depth-map comparison at a ratio of 177:1 with the method of Hoffmann et al. [127]. In both cases, the specialised PDE-based methods outperform JPEG2000. Images were published in consent with the corresponding authors.

> "The following list of functionality and goals can be useful in separating data compression techniques that have niche opportunity from general-purpose methods. Although some of the DCT-specific requirements may not apply, any data compression method that might be considered competitive to JPEG needs to be evaluated on the basis of all of the general requirements, not merely on its own strongest features." [215, p. 304]

The quote above shows that Pennebaker and Mitchell were already aware that new compression concepts require evaluation categories that are independent of the specific challenges of transform-based compression. In the spirit of the original JPEG design goals, we propose the following five properties to assess the viability of a codec.

Quality. For a given file size, the codec should yield a reconstruction that is as close to the original as possible. Preferably, this should be evaluated quantitatively with an adequate error measure.

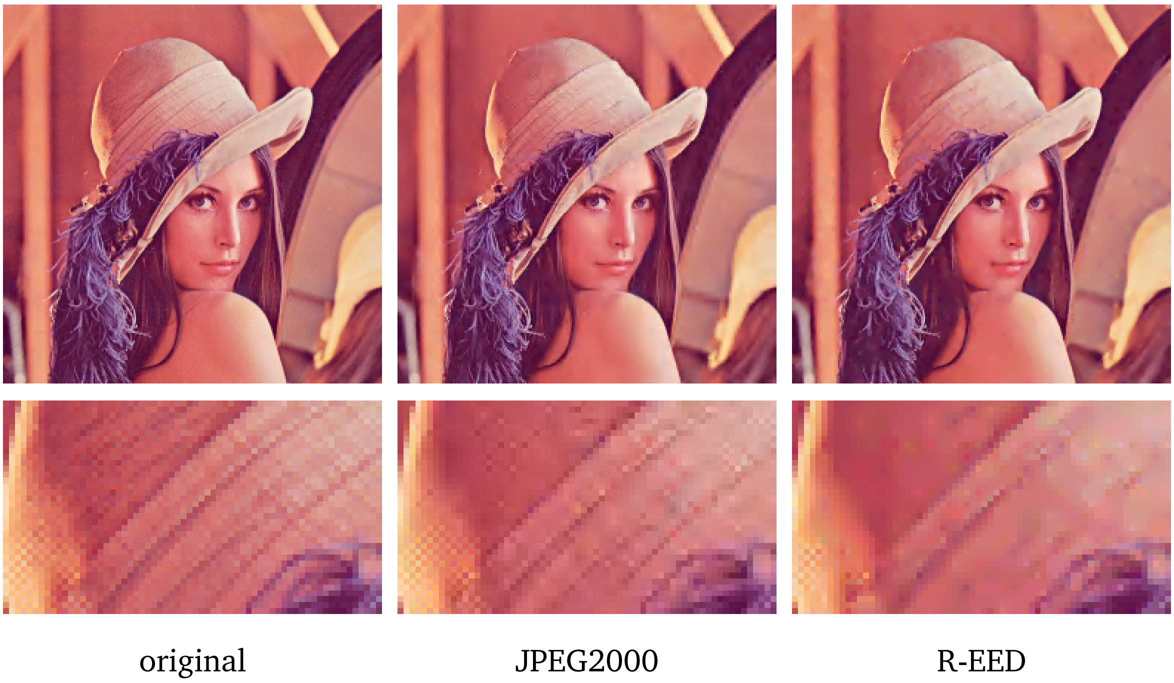


Figure 1.3.: **Comparison of R-EED and JPEG2000 on the colour image *lena*.** R-EED is not able to surpass JPEG2000 quality-wise at a compression ratio of 20:1. This is particularly obvious in the zoom-ins of the second row.

Consistency. For a general-purpose method, reconstruction quality should vary as little as possible with image content. Codecs that only perform well for limited content type might still have merit as specialised methods.

Perceptive Coding. Compression codecs designed for humans should exploit the human visual system and remove data that is perceived as redundant. Especially, dedicated processing of colour images should be included. Pennebaker and Mitchell [215] formulate this goal specifically with respect to chroma subsampling, a technique for storing colour information less accurately than brightness information. In contrast, our definition is more general: We consider all methods that exploit human perception.

Progressiveness. A codec should be able to provide the option to reconstruct lower quality versions of an image from only partially given compressed files. This is an important property for the use of codecs in digital communication (e.g. transmission over the internet).

Immediacy. The codec should have a runtime that is adequate on consumer hardware for the given task. Broadcasting codecs (e.g. for TV broadcast or Blu-Ray distribution) can have a long compression time, since this is usually done only once. For decoding, real-time performance is a requirement, especially for video decoding. General purpose codecs for everyday use are also expected to be *synchronous*, i.e. there should not be a big discrepancy between encoding and decoding time.

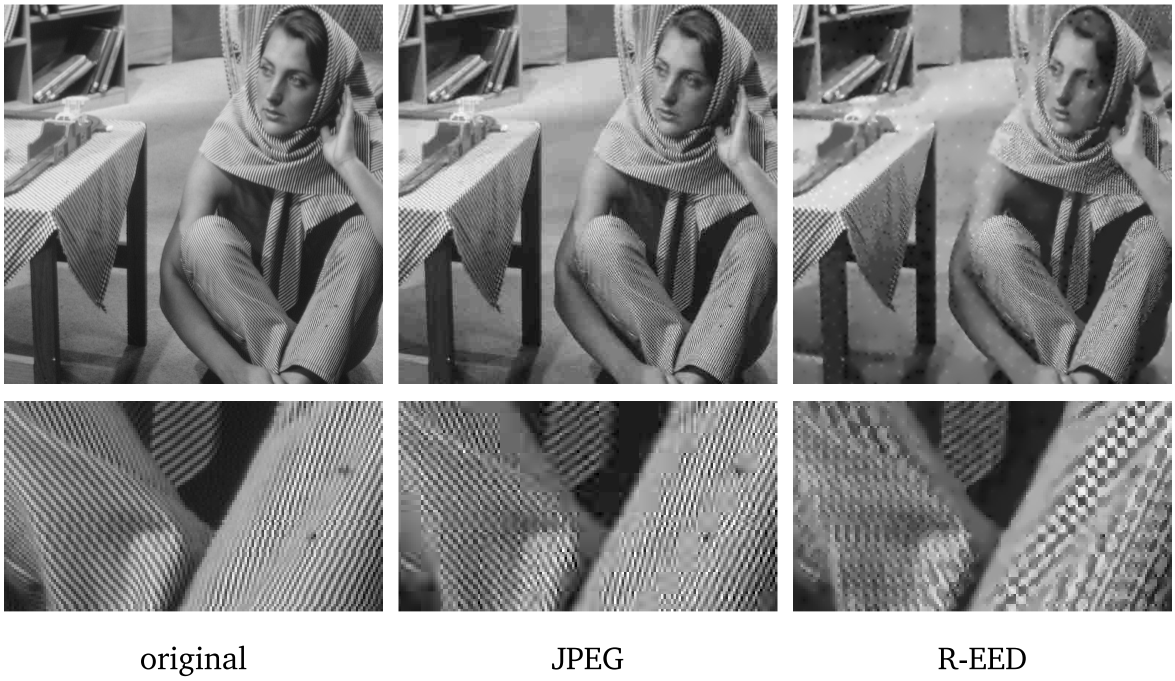


Figure 1.4.: **Comparison of R-EED and JPEG on the test image *barbara*.** As the zoom-ins in the bottom row show, R-EED does not suffer from block artefacts as JPEG does. However, JPEG still outperforms R-EED at a compression ratio of 18:1, since it can reconstruct fine scale textures.

Currently, R-EED is the most successful PDE-based codec for general image content. In the following, we assess in how far it is competitive to transform-based compression according to the aforementioned criteria. As Figure 1.1 shows, the quality requirement is currently fulfilled on grey value images. The codec is, however, not consistent due to its weaknesses in regard to texture images as shown in Figure 1.4. Perceptive coding is not implemented in R-EED, which explains the unfavourable comparison to JPEG2000 on colour images in Figure 1.3. R-EED is already progressive due to an extension of R-EED by Schmaltz et al. [251]. However, it does not fulfil immediacy requirements, since optimal results require hours of encoding on consumer hardware and decompression does not reach real-time, yet. There is a modification of R-EED for fast en- and decoding [182] in the context of steganography and two broadly related real-time video approaches [22, 156]. However, all of these sacrifice quality for speed.

In summary, the observations above indicate that R-EED can currently compete with JPEG and JPEG2000 in two of five categories. This alone is an impressive feat considering the large research and engineering community behind transform-based compression. However, the missing three categories also imply that there are still a lot of unsolved problems in PDE-based compression. This thesis aims at exploring the hitherto unknown potential of general purpose encoding with PDEs under practical constraints.

1.1 Goals and Contribution

> *The first step towards getting somewhere is to decide that you are not going to stay where you are.*
>
> — **John P. Morgan**
> (Entrepreneur)

This thesis is dedicated to two overarching goals: On the one hand, it aims to foster understanding of the strengths and weaknesses of PDE-based compression. On the other hand, we want to use these insights to introduce new features to PDE-based compression. These features should contribute to a competitive PDE-based codec in the sense of the five requirements from the previous section: quality, consistency, perceptive coding, progressiveness, and immediacy.

Understanding PDE-based Coding

As mentioned in the previous section, the most successful encoders rely on EED for the reconstruction of missing image parts, the so-called *inpainting*. In order to understand this success, we have to consider the interaction of several main ingredients for PDE-based compression. First, finding a suitable diffusion operator for the reconstruction is vital. Next, one must decide which known data is kept. Finally, this data must be stored efficiently. In order to get a better understanding of PDE-based coding, we investigate three different questions:

1. How can we explain the success of anisotropic diffusion in image compression?

2. What different possibilities are there to select and store known data, and which of them is the most promising?

3. How do the inpainting capabilities of different PDEs interact with other compression steps and what does this imply for selecting the right inpainting PDE?

To find answers to these questions, we start by analysing diffusion models for denoising and inpainting. Instead of interpreting diffusion from the point of view of physical propagation models, which is common in PDE-based compression, we consider a probabilistic approach. To this end, we investigate relations between Bayesian and variational methods for denoising. Our systematic analysis of connections between prior information from natural image databases and diffusion models leads to a unifying framework for tensor-driven diffusion. From the structure of the prior information and experiments on large databases we can explain the success of anisotropic diffusion models. In addition, we also explore learning techniques for PDE-based compression.

Our next step is the investigation of different techniques to store data in image compression. The overwhelming majority of diffusion-based codecs stores a fraction of image points as known data. Therefore, these methods have to store both the position and value of these pixels. For the grey or colour values, we consider coarse quantisation, i.e. sampling in the co-domain, and different entropy encoders that remove redundancy. We also explore different ways to represent and store positions. In particular, we compare the cost of free, pixel-accurate positions to known data that is restricted to a locally adaptive grid (as in R-EED).

Finally, we combine our findings from the two last contributions and deal with the interdependency of inpainting operators and compression steps. We determine how restrictions of point selection and coarse quantisation influence the quality of the three most popular inpainting techniques in PDE-based compression: homogeneous, biharmonic, and edge-enhancing diffusion. For the selection of known data, we compare an optimal control approach by Hoeltgen et al. [124] to a new probabilistic method for restricted point sets. In addition, we also evaluate both methods against the heuristic strategies on restricted point sets that are employed by R-EED. Together with the insights on efficient storage of known data with entropy encoders, we show the importance of evaluating inpainting operators in combination with a full compression codec. Their performance on sparse known data does not necessarily reflect the trade-off between file size and quality accurately, since the compression aspect is neglected in such comparisons.

New Features for PDE-based Coding

Out of the five different requirements from the previous section, we consider the three unsolved ones: consistency, perceptive coding, and immediacy.

In regard to *consistency*, we focus on heavily textured images. We first establish that EED is particularly well-suited for the reconstruction of coarse image structures, but struggles with fine scale texture. This implies the need to combine PDE-based inpainting with other techniques that can deal with the reconstruction of high-frequent patterns. To this end, we consider an exemplar-based inpainting approach for sparse known data by Facciolo et al. [90]. Reconstruction methods of this type rely on the similarities between regions surrounding the unknown pixels, so-called patches, to known image areas. Such patch-based approaches can reconstruct especially regular patterns accurately. We propose a block-based approach that efficiently combines both inpainting methods into one codec with minimal overhead. This codec is designed in such a way that other texture inpainting methods can be easily incorporated.

As a new feature for *immediacy*, we propose a real-time video decoding framework based on R-EED. For demonstration purposes, we design an extendible video codec that does not use redundancy between subsequent frames to improve the compression ratio, yet. Instead, we aim at making decompression possible at 25 frames per second while preserving the quality of R-EED on every single frame. This is achieved by a combination of efficient solvers, parallelisation on both CPU and GPU, as well as specifically designed initialisation techniques that reuse information from previous frames. In our experiments we show that the reconstructed frames in real-time playback are very close to reference reconstructions without time constraints.

Moreover, we introduce two new features for *perceptive coding* to PDE-based compression. First, we propose a straightforward, efficient model for region of interest coding (ROI) which can be applied to all of our previous codecs. Our ROI framework enables the user to assign a quality weight to each pixel before compression. During compression, the codec adapts the reconstruction quality locally. Such features are for instance required by medical applications. Furthermore, we introduce a colour compression mode that exploits the fact, that the human visual system has a higher tolerance for reconstruction errors in colour information than in structure information. First, we perform a decomposition of the image into brightness and colour data with a YCbCr colour space. Then, we propose new colourisation techniques that use the structure information of the brightness channel to guide the diffusion in the colour channels. By storing more known data for the brightness channel than for the colour channels, we can guarantee an accurate reconstruction of the brightness channel by regular diffusion inpainting. We can reuse this structure information to decompress the whole image with our colourisation techniques. In our experiments we show that this colour coding approach not only improves the image visually, it also allows us to surpass JPEG and JPEG2000 regarding quantitative error measures.

1.2 Organisation of the Thesis

> ” *Style and structure are the essence of a book;*
> *great ideas are hogwash.*
>
> — **Vladimir Nabokov**
> (Writer)

After the first chapter, which motivates this thesis and establishes the goals and contributions, we address notations and related work in Chapters 2 and 3. We discuss both the history of transform-based compression and PDE-based compression, and also briefly address other approaches. Since all chapters of this thesis rely on diffusion inpainting, we dedicate one section to the analytical and numerical background of these processes. Many important ideas from R-EED influence the codecs in this thesis, which is why we also discuss this compression method in detail.

The core contributions of this thesis are heavily interconnected, but can also stand on their own since they treat different aspects of PDE-based compression. We have organised them in chapters that can each be attributed to one of the goals from Section 1.1. These core contributions can be read in arbitrary order and contain notes on their connections to other chapters.

Chapters 4 and 5 both contribute to deeper insights into the core ideas of PDE-based compression. In Chapter 4, we motivate a unifying framework for tensor-driven diffusion from the statistics of natural images and analyse its implications for compression. We assess the representation and storage of known data in Chapter 5 and investigate the influence of both factors on the choice of diffusion operators. This chapter can also be associated with the quality aspect from the requirements of Section 1.1.

New features are introduced in the Chapters 6, 7, and 8. We cover a hybrid approach that combines PDE-based interpolation with patch-based inpainting in Chapter 6. This block-based method is designed to foster consistency on textured images. A framework for real-time video playback is introduced in Chapter 7, thus advancing PDE-based compression in regard to immediacy. Finally, we address perceptive coding by proposing novel ROI and colour compression features in Chapter 8.

The main part of this thesis closes with conclusions and an outlook on future work in Chapter 9. In addition to the description above, Figure 1.5 visualises the structure of this thesis.

At the end of this document, several useful additional sources of information can be found: the bibliography, an index, and lists of own publications, acronyms, and mathematical operators. Furthermore, there are appendices that offer additional detailed information that was omitted in the main chapters in order to increase readability. Appendix G contains detailed proofs for the probabilistic models of Chapter 4. Details about external code used for this thesis can be found in Appendix H. Finally, in Appendix I, we propose a sketch for a common framework for PDE-based compression codecs as discussed in Chapter 9.

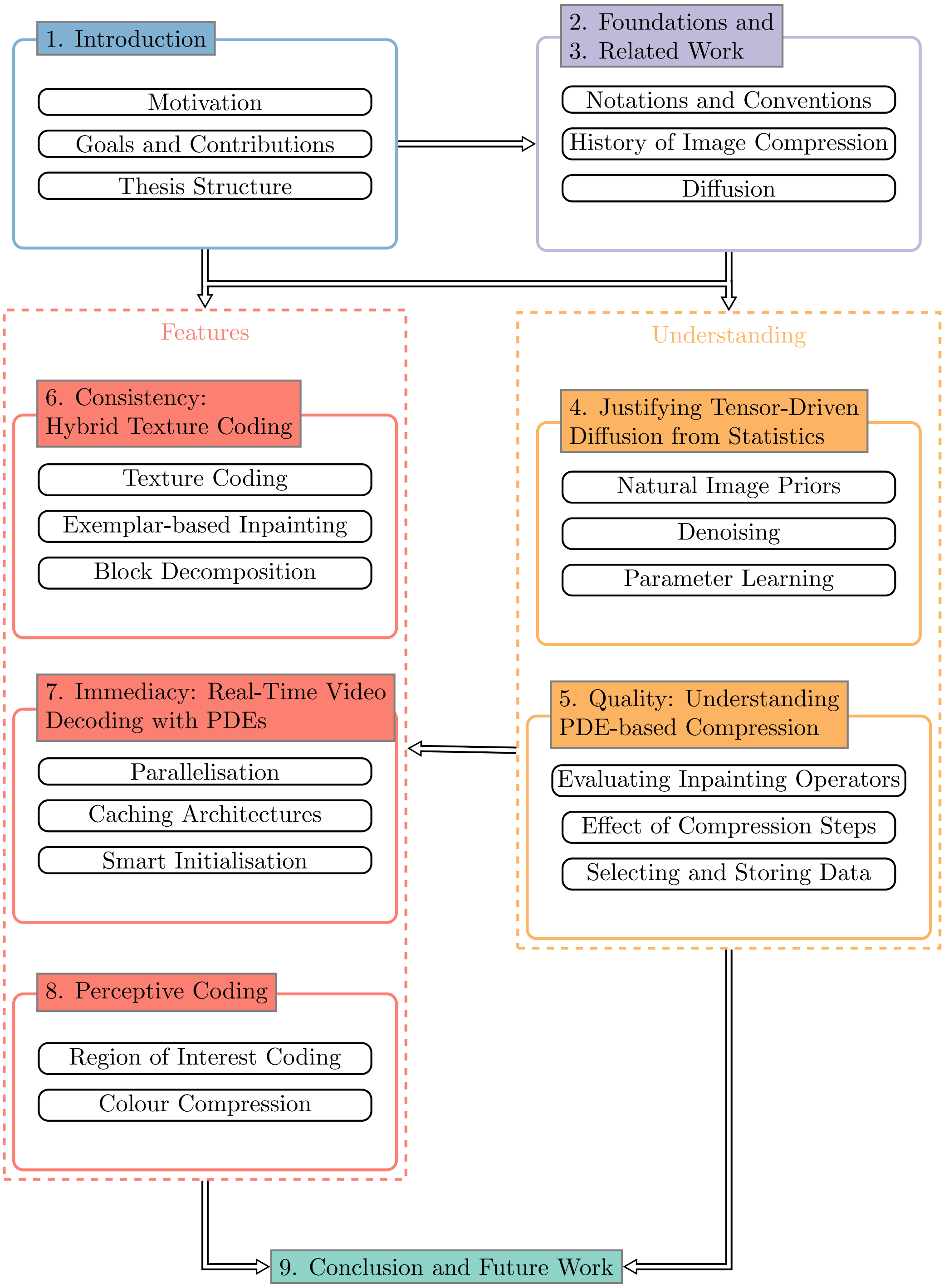

1. Introduction
Motivation
Goals and Contributions
Thesis Structure
2. Foundations and
3. Related Work
Notations and Conventions
History of Image Compression
Diffusion
Features
6. Consistency: Hybrid Texture Coding
Texture Coding
Exemplar-based Inpainting
Block Decomposition
7. Immediacy: Real-Time Video Decoding with PDEs
Parallelisation
Caching Architectures
Smart Initialisation
8. Perceptive Coding
Region of Interest Coding
Colour Compression
Understanding
4. Justifying Tensor-Driven Diffusion from Statistics
Natural Image Priors
Denoising
Parameter Learning
5. Quality: Understanding PDE-based Compression
Evaluating Inpainting Operators
Effect of Compression Steps
Selecting and Storing Data
9. Conclusion and Future Work

Figure 1.5.: **Visual Thesis Outline.** Arrows visualise connections between different chapters. The main contributions can be grouped into two categories. Chapters from the category *understanding* provide fundamental insights that influence the design of new *features*.

2 Definitions, Notations, and Conventions

> *There should be no such thing as boring mathematics.*
>
> — **Edsger W. Dijkstra**
> (Computer Scientist)

This chapter introduces the most important mathematical notations and definitions, as well as some non-mathematical formatting decisions that are relevant for this thesis. Since such formal conventions are inherently dry, but necessary nonetheless, we keep this chapter as brief as possible.

Section 2.1 deals with didactically motivated visual cues that guide the reader throughout the whole thesis. The basic mathematical definitions and conventions in Sections 2.2 and 2.3 follow fairly common standards in image compression literature and can be skipped by experienced readers. The chapter closes with a definition and discussion of different error measures, namely the mean squared error (MSE) and structural similarity measure (SSIM).

2.1 Formatting and Visual Cues

> *I believe in a visual language that*
> *should be as strong as the written word.*
>
> — **David LaChapelle**
> (Photographer)

This thesis contains different kinds of visual cues that help to structure the content and ease navigation through the document. In the following, you find self-explanatory examples for such formatting tools.

Example: Key Idea

Key idea boxes summarise a concept in one or two short sentences to capture the essence of a given paragraph or chapter.

Example: Publication Box

Publication boxes make transparent where certain parts of the thesis are published and if co-authors have contributed to a certain topic. A publication box can be found at the end of each self-contained section. Note that Appendix H contains a separate list of implementations by other researchers that were used in this thesis.

Example: Reference Box

☞ In *reference boxes* such as this one, important connections to other parts of the thesis are presented. This includes e.g. in which chapter newly introduced theoretical concepts are applied or references to more detailed information in an appendix.

2.2 Basic Notations and Definitions

> *It is worth noting that the notation facilitates discovery. This, in a most wonderful way, reduces the mind's labour.*
>
> — **Gottfried Wilhelm von Leibniz**
> (Mathematician and Philosopher)

The first important convention that we apply to ease the readability of formulas is the use of capitalisation and bold face to indicate different mathematical entities. This makes it easy to distinguish scalar values, functions, vectors, matrices, and sets from each other.

Scalars and vectors both use lower-case letters. However, vectors are additionally marked by bold face, e.g. $\boldsymbol{v} = (v_1, \ldots, v_n) \in \mathbb{R}^n$ is a real-valued vector with $n \in \mathbb{N}$ scalar components v_i. We also denote functions by lower-case letters and apply bold face depending on their co-domain. Therefore, vector-valued functions can be distinguished from scalar-valued ones by their boldface typesetting. Functionals that map for example infinitely often differentiable functions u from the class C^∞ to a real value are denoted by capital letters without bold face, i.e. $F : C^\infty \to \mathbb{R}$. Note that in some rare cases we deviate from the capitalisation conventions for functions if they contradict commonly used naming conventions.

Matrices and sets are represented by capitalised letters. Matrices are marked additionally in bold face. For example, $\boldsymbol{M} \in \mathbb{R}^{m \times n}$ denotes a real-valued $m \times n$ matrix. Tab. 2.1 also provides a quick overview over these conventions.

Table 2.1.: Notational Conventions for Different Mathematical Entities.

entity	capitalisation	bold face	examples
scalar value	no	no	$a, \sigma \in \mathbb{R}$
vector	no	yes	$\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$
matrix	yes	yes	$\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times m}$
set	yes	no	$\Omega \subset \mathbb{R}$
function	no	depends	$f : \mathbb{R}^2 \to \mathbb{R}$, $\boldsymbol{g} : \mathbb{R}^2 \to \mathbb{R}^3$
functional	yes	no	$F : C^\infty \to \mathbb{R}$

Basic Operators

Many important definitions later on require the notion of dot products and norms, e.g. directional derivatives. If not stated otherwise, dot products always refer to the Euclidean dot product and vector norms to the induced Euclidean norm:

$$\forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n : \quad \langle \boldsymbol{x}, \boldsymbol{y} \rangle := \boldsymbol{x}^\top \boldsymbol{y} = \sum_{i=1}^{n} x_i y_i, \tag{2.1}$$

$$|\boldsymbol{x}| := \sqrt{\boldsymbol{x}^\top \boldsymbol{x}} = \sqrt{\sum_{i=1}^{n} x_i^2}. \tag{2.2}$$

Moreover, the operator $|\cdot|$ is used for the absolute value of scalars. Finally, we also define it for sets: If a set K is finite, $|K|$ denotes number of its elements, i.e. the *cardinality*. For sets with an infinite number of elements, e.g. $\Omega \subset \mathbb{R}$, we integrate over the corresponding area instead:

$$|\Omega| := \int_\Omega 1 \, dx. \tag{2.3}$$

This is often used for averaging a quantity over the whole set, e.g. for computing the average grey value of an image.

The dot product defined in Eq. 2.1 also allows us to introduce our notation for orthogonality:

$$\boldsymbol{x} \perp \boldsymbol{y} \quad \Leftrightarrow \quad \boldsymbol{x}^\top \boldsymbol{y} = 0, \tag{2.4}$$

$$\boldsymbol{x}^\perp \perp \boldsymbol{x}. \tag{2.5}$$

Orthogonal vectors play a large role in defining anisotropic diffusion processes in Section 3.1.

Another important operator that is closely related to diffusion is convolution. For two functions $f, g : \mathbb{R}^n \to \mathbb{R}$ it can be interpreted as blending the function f with the mirrored function g:

$$(f * g)(\boldsymbol{x}) := \int_{\mathbb{R}^n} f(\boldsymbol{y}) g(\boldsymbol{x} - \boldsymbol{y}) \, d\boldsymbol{y}. \tag{2.6}$$

Linear system theory tells us that every linear shift-invariant filter can be expressed as a convolution [49], and this holds also true for linear diffusion discussed later on. In order to establish this connection we will need *Gaussian* kernels. For $\boldsymbol{x} \in \mathbb{R}^n$, an n-D Gaussian with standard deviation σ and mean 0 is defined by

$$K_\sigma(\boldsymbol{x}) := \frac{1}{(\sqrt{2\pi\sigma^2})^n} \exp\left(-\frac{|\boldsymbol{x}|^2}{2\sigma^2}\right). \tag{2.7}$$

Gaussian convolution is also an invaluable tool for presmoothing, i.e. removing unwanted high-frequent perturbations from an image with low computational cost.

Finally, for compression algorithms such as arithmetic coding or in the context of image discretisation, we require the floor function

$$\lfloor x \rfloor := \max_{k\in\mathbb{Z},\, k\leq x} k. \tag{2.8}$$

Differentiation

All contributions in this thesis deal with partial differential equations. Consequently, *partial derivatives* occur regularly. First of all, we assume all of the functions that we consider in the following to be sufficiently often differentiable. We choose several equivalent notations depending on the context in which they appear. For a function $f : \mathbb{R}^n \to \mathbb{R}$, we denote the corresponding partial derivative at a point $\boldsymbol{x} \in \mathbb{R}^n$ in respect to the i-th argument of f by

$$\frac{\partial}{\partial x_i} f(\boldsymbol{x}) = \partial x_i f(\boldsymbol{x}) = f_{x_i}(\boldsymbol{x}) := \lim_{h\to 0} \frac{f(x_1, \ldots, x_i + h, \ldots, x_n) - f(\boldsymbol{x})}{h}. \tag{2.9}$$

In addition to partial derivatives, the *gradient* $\boldsymbol{\nabla} f$, the *divergence* div$\boldsymbol{f}$, and the *Laplace operator* Δf are the most important differential operators. They are defined as

$$\boldsymbol{\nabla} f(\boldsymbol{x}) := \left(\frac{\partial}{\partial x_1} f(\boldsymbol{x}), \ldots, \frac{\partial}{\partial x_n} f(\boldsymbol{x})\right)^T, \qquad f : \mathbb{R}^n \to \mathbb{R}, \tag{2.10}$$

$$\mathrm{div}\boldsymbol{f}(\boldsymbol{x}) := \sum_{i=1}^{n} \frac{\partial}{\partial x_i} f_i(\boldsymbol{x}), \qquad \boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n, \tag{2.11}$$

$$\Delta f(\boldsymbol{x}) := \mathrm{div}(\boldsymbol{\nabla} f) = \sum_{i=1}^{n} \partial_{x_i x_i} f(\boldsymbol{x}), \qquad f : \mathbb{R}^n \to \mathbb{R}. \tag{2.12}$$

It is important to note at this point that, by convention, these operators are purely *spatial* if not mentioned otherwise. This means that for a function $f(\boldsymbol{x}, t) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ that does depend on a spatial coordinate $\boldsymbol{x}$ and a time coordinate t, the operators defined above only consider the partial derivative in space, not in time.

For certain anisotropic models that we discuss in this thesis, *directional derivatives* are required in addition to partial derivatives. Formally, for a function $f : \mathbb{R}^n \to \mathbb{R}$ the derivative in the direction of a vector $\boldsymbol{v} \in \mathbb{R}^n$ is defined by

$$\partial_{\boldsymbol{v}} f(\boldsymbol{x}) := \lim_{h \to 0} \frac{f(\boldsymbol{x} + h\boldsymbol{v}) - f(\boldsymbol{x})}{h}. \tag{2.13}$$

If the limit above exists, it is also possible to express the directional derivative by the dot product

$$\partial_{\boldsymbol{v}} f(\boldsymbol{x}) = \boldsymbol{v}^\top \boldsymbol{\nabla} f(\boldsymbol{x}). \tag{2.14}$$

The derivation of diffusion processes from continuous energy functionals which map functions to a real valued energy requires yet another class of derivatives. Gâteaux derivatives are a generalisation of differential calculus to functional derivatives. That is, let F be a functional that maps functions $\boldsymbol{u} : \mathbb{R} \mapsto \mathbb{R}$ to real values. Then, the Gâteaux derivative is defined by

$$d_u F(u) := \lim_{\epsilon \to 0} \frac{F(u + \epsilon h) - F(u)}{\epsilon} \tag{2.15}$$

with a test function $h : \mathbb{R}^2 \mapsto \mathbb{R}$ and $\epsilon \in \mathbb{R}$. For our purposes it is often more convenient to interpret the Gâteaux derivative as a derivative in respect to ϵ that is evaluated at $\epsilon = 0$:

$$d_u F(u) = \frac{d}{d\epsilon} F(u + h\epsilon)\bigg|_{\epsilon=0}. \tag{2.16}$$

In some cases, the basic definitions for the different kinds of derivatives above need to be extended to e.g. vectorial functions. In this case, definitions are provided in the respective chapters.

Note that Appendix D contains a list of symbols that occur frequently in this thesis.

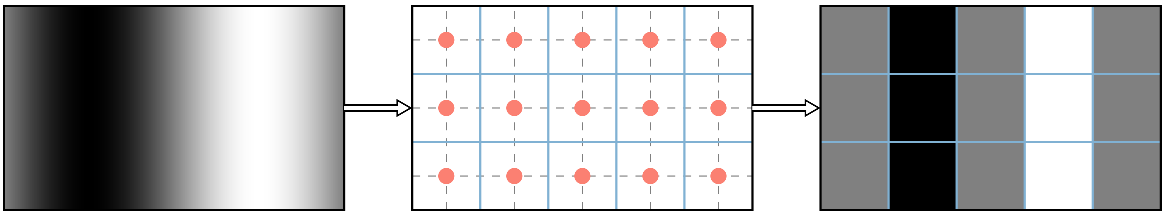


Figure 2.1.: **Sampling of 2-D Image.** The sketch above depicts the sampling of a continuous image $f : \Omega \to \mathbb{R}$ to a discrete $\boldsymbol{f} \in \mathbb{R}^{3\times5}$ image. The red dots mark the sampling points on the regular grid represented by blue dashed lines.

2.3 Images

> *The essential fact is that all the pictures which science now draws of nature, and which alone seem capable of according with observational facts, are mathematical pictures.*
>
> — **Sir James Jeans**
> (Physicist and Mathematician)

In our work we deal with two-dimensional grey scale or colour images. For each point in a spatial 2-D coordinate system, the image provides a grey or colour value. Therefore, we consider images to be continuous functions

$$\boldsymbol{f} : \Omega \to \mathbb{R}^{n_c}. \tag{2.17}$$

They map coordinates from the rectangular *image domain* $\Omega = [a, b] \times [c, d]$ to the co-domain $\mathbb{R}^{n_c}$ with $n_c \in \mathbb{N}$. In particular, grey scale images $(n_c = 1)$ map to real numbers that represent a certain grey value and colour images $(n_c > 1)$ map to vectors from a given colour space. In most cases, the input data relies on an RGB colour space where $\boldsymbol{f}(x, y) = (f_R(x, y), f_G(x, y), f_B(x, y))$ specifies a red, green, and blue component for each image point. In Chapter 8 we also employ a YCbCr colour space that separates brightness and colour information into different channels.

This continuous formulation is important for our physically motivated diffusion PDEs. Throughout the whole thesis we always assume that our image functions are sufficiently often differentiable.

In practice, the images that one obtains from digital image acquisition methods are discrete. A digital camera for example possesses a finite array of sensors that record light photons. These measurements are therefore only available on a subset of the continuous image domain Ω.

Thus, a digital image with *resolution* $n_x \times n_y$ is actually a mapping from a discrete grid $\Gamma := \{0, \ldots, n_x - 1\} \times \{0, \ldots, n_y - 1\}$ to a grey or colour value range. Such a discretisation of

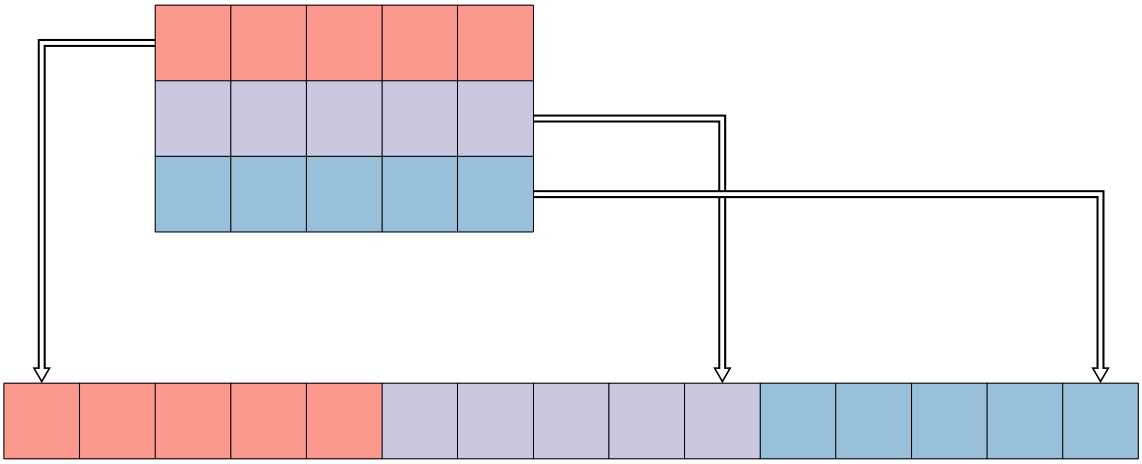


Figure 2.2.: **Vectorisation of a 2-D Image.** The sketch above depicts the mapping of a 2-D image $\boldsymbol{f} \in \mathbb{R}^{3\times5}$ to a vector $\boldsymbol{f} \in \mathbb{R}^{15}$ according to Eq. 2.20. The rows of the image are simply appended from top to bottom and form a single vector.

the spatial domain is known as *sampling*. Here we assume that samples are taken on Ω with constant *grid sizes* h_x, h_y:

$$\forall\, i,j \in \Gamma : \; \boldsymbol{f}_{i,j} := f\left(a + \left(\frac{1}{2} + i\right) h_x, c + \left(\frac{1}{2} + j\right) h_y\right) \tag{2.18}$$

$$n_x \cdot h_x = b - a, \quad n_y \cdot h_y = d - c. \tag{2.19}$$

This sampling can be interpreted as subdividing the rectangular domain Ω into $n_x \times n_y$ pixels of size $h_x \times h_y$ and assigning each pixels the value of the continuous image f at its centre. This process is visualised by Figure 2.1.

Note that we deviate slightly from our notational conventions here, since formally, $\boldsymbol{f} \in \mathbb{R}^{n_y \times n_x}$ could be seen as a matrix and thus should be denoted by a capital letter. However, we consider images as vectors $\boldsymbol{f} \in \mathbb{R}^{n_x n_y}$ instead. Each 2-D image can be transformed into a vector by appending all rows successively one after another as demonstrated in Figure 2.2. Formally, this comes down to the index transformation

$$v_\alpha = v_{i,j} \quad \text{with} \quad j := \lfloor \alpha / n_y \rfloor + 1,\; i := \alpha - j n_y, \tag{2.20}$$

$$\text{i.e.} \quad \alpha := (j-1)n_y + i. \tag{2.21}$$

We exploit notation here and allow to use both the vectorial or matrix interpretation of $\boldsymbol{f}$ wherever it is convenient. For example, it can be intuitive to describe 2-D neighbourhoods with the 2-D index. However, the vector notation can be very convenient if for example a filter application is written as a matrix vector product.

After sampling, the image is only semi-discrete, i.e. the co-domain is still continuous. However, digital images that are stored e.g. on a hard disk are usually fully discrete. We assume that the co-domain of our input images contains grey or colour values from the set $\{0, 1, \ldots, 255\}$

which corresponds to an 8bit representation. The discretisation of the co-domain is called *quantisation* and is not necessarily limited to the range $\{0, \ldots, 255\}$.

☞ **Quantisation in Compression**

For compression, quantisation plays an important role since it reduces file size at the cost of accuracy. In Chapter 5, we discuss the effects of quantisation on human perception. Furthermore, we investigate how coarse quantisation affects the reconstruction from sparse known data with different diffusion-PDEs.

2.4 Error Measures

> *In science, one man's noise is another man's signal.*
>
> — **Edward Ng**
> (Mathematician and Engineer)

Since the task of image compression is to reduce the file size of an image while staying visually close to the original, a way to judge the quality of a compressed image is necessary. In image coding, the original, uncompressed image is considered to be the ground truth. Since such a ground truth is available, it makes sense to use *full reference metrics* [273] that compare a processed image to the reference. In classic compression, quantitative measures like the mean squared error (MSE) or peak signal-to-noise ratio (PSNR) are the standard. However, a perceptually motivated error metric, the structural similarity measure (SSIM), has received a lot of attention in the image processing community with the original paper [289] being cited more than 10000 times at the time of writing and several practical applications dedicated to SSIM [175, 243, 291]. Therefore, a discussion and comparison of error measures is inevitable in this thesis.

One of the most popular quality measures in image compression is still the MSE. It represents the deviation of the coded image from the original in terms of the average squared per-pixel distance. One big advantage is its simplicity:

$$\mathrm{MSE}(\boldsymbol{f}, \boldsymbol{g}) := \frac{1}{|\Gamma|} \sum_{(i,j) \in \Gamma} \Big(f_{i,j} - g_{i,j} \Big)^2 . \tag{2.22}$$

Obviously, the MSE is 0 if the images are equal and increases the more the images deviate from each other. It is not bounded from above if the co-domain of the images is unlimited. The

MSE is also a metric in the mathematical sense, i.e. it fulfils the following four properties for arbitrary images $\boldsymbol{f}$, $\boldsymbol{g}$, and $\boldsymbol{h}$ with the same resolution:

$$\mathrm{MSE}(\boldsymbol{f},\boldsymbol{g}) \geq 0 \tag{2.23}$$

$$\mathrm{MSE}(\boldsymbol{f},\boldsymbol{g}) = 0 \Leftrightarrow \boldsymbol{f} = \boldsymbol{g} \tag{2.24}$$

$$\mathrm{MSE}(\boldsymbol{f},\boldsymbol{g}) = \mathrm{MSE}(\boldsymbol{g},\boldsymbol{f}) \tag{2.25}$$

$$\mathrm{MSE}(\boldsymbol{f},\boldsymbol{h}) \leq \mathrm{MSE}(\boldsymbol{f},\boldsymbol{g}) + \mathrm{MSE}(\boldsymbol{g},\boldsymbol{h}) \tag{2.26}$$

In image denoising, a modified version of the MSE, the PSNR is very popular, but also appears in literature on compression. Essentially, it defines a logarithmic rescaling of the MSE by

$$\mathrm{PSNR}(\boldsymbol{f},\boldsymbol{g}) := 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}(\boldsymbol{f},\boldsymbol{g})}\right). \tag{2.27}$$

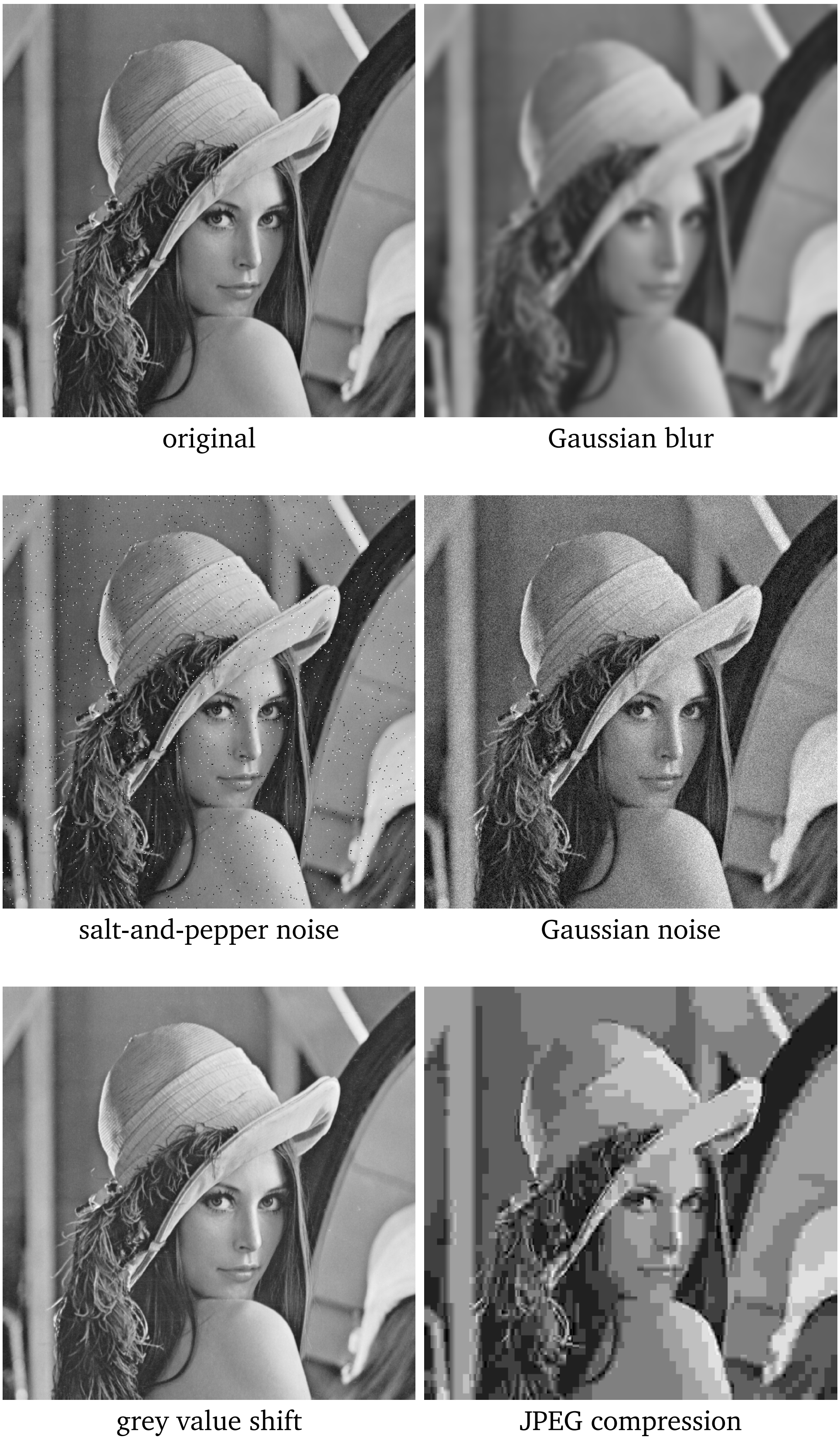
Here we assume that the image's co-domain is bounded from above by 255. In contrast to the MSE, large PSNR values are better since it approaches infinity the closer the coded image gets to the ground truth. Since MSE and PSNR are equivalent, we can choose freely which of them to use. Usually, we prefer the MSE in this thesis except for the few occurrences of denoising in Chapter 4, where we use the PSNR instead.

Unfortunately, the MSE does not always accurately model human perception. Wang et al. [289] came forward with examples that show different distortions of the same image that lead to a similar MSE but different perceived quality. For such collections of images that produce equal results in a full reference metric, Dosselmann and Yang [78] use the term *hypersphere*.

Figure 2.3 shows an MSE hypersphere with the following distortions: Gaussian blur (convolution with a Gaussian), salt-and-pepper noise (a given percentage of error pixels is set to black or white), Gaussian noise (noise that follows a Gaussian distribution), grey value (each grey value is increased by the same constant), and JPEG compression. For each distortion, the respective parameters were tuned to yield an MSE of approximately 200. It is obvious that some of these degradations are much more prominent than others. For example, a human might perceive almost no deterioration for the grey value shift, but a severe drop in quality for the blocking artefacts of JPEG.

As a remedy for this problem, Wang et al. [289] proposed a new quality metric, the so-called SSIM. Its definition is significantly more sophisticated than the simple MSE. The SSIM relies on the assumption that the human visual system perceives an image as high-quality, if the structural information in natural images, i.e. the dependencies of neighbouring pixels, are accurately represented.

In the following, we discuss the three components that, together with an appropriate weighting, form the SSIM. All of these measurements are conducted locally on small image patches and



original

Gaussian blur

salt-and-pepper noise

Gaussian noise

grey value shift

JPEG compression

Figure 2.3.: **MSE Hypersphere.** Collection of images with various degradations and an MSE of approximately 200.

apply Gaussian weights $\boldsymbol{w}$ according to the distance from the patch centre. First, let the weighted mean value on a patch $P \subset \Gamma$ of the image $\boldsymbol{f}$ be defined as

$$\mu_{P,\boldsymbol{f}} := \frac{1}{|P|} \sum_{(i,j) \in P} w_{i,j} f_{i,j}. \tag{2.28}$$

Based on this mean, the authors propose a *luminance* component that should represent the local relative luminance changes of objects in the scene. Please note that for the sake of consistency to the original SSIM notations we use the term "luminance" in this chapter even though we do not deal with absolute photometric quantities in this thesis. We consider abstract gray or brightness values without any consideration of physical implications from image acquisition or display.

In practice, the investigation of luminance changes comes down to a comparison of the local means of two images $\boldsymbol{f}$ and $\boldsymbol{g}$ according to

$$\ell(P, \boldsymbol{f}, \boldsymbol{g}) = \frac{2\mu_{P,\boldsymbol{f}}\mu_{P,\boldsymbol{g}} + \alpha}{\mu_{P,\boldsymbol{f}}^2 + \mu_{P,\boldsymbol{g}}^2 + \alpha} \tag{2.29}$$

where α is a free parameter. For the next component, the authors propose to remove the local mean luminance from the image in order to compare contrast. To this end, they employ the weighted standard deviation of a patch P on an image $\boldsymbol{f}$ which is given by

$$\sigma_{P,\boldsymbol{f}} := \sqrt{\frac{1}{|P|-1} \sum_{(i,j) \in P} w_{i,j} (f_{i,j} - \mu_{P,\boldsymbol{f}})^2}. \tag{2.30}$$

In a similar way as in Eq. 2.29, the authors define the *contrast* component with a free parameter β as

$$c(P, \boldsymbol{f}, \boldsymbol{g}) = \frac{2\sigma_{P,\boldsymbol{f}}\sigma_{P,\boldsymbol{g}} + \beta}{\sigma_{P,\boldsymbol{f}}^2 + \sigma_{P,\boldsymbol{g}}^2 + \beta}. \tag{2.31}$$

Finally, the structural similarity is measured via the correlation coefficient between two images $\boldsymbol{f}$ and $\boldsymbol{g}$ on a patch P:

$$\sigma_{P,\boldsymbol{f},\boldsymbol{g}} := \frac{1}{|P|-1} \sum_{(i,j) \in P} w_{i,j} (f_{i,j} - \mu_{P,\boldsymbol{f}})(g_{i,j} - \mu_{P,\boldsymbol{g}}). \tag{2.32}$$

Again, the structure comparison term can then be defined in a similar fashion as the previous two components by

$$s(P, \boldsymbol{f}, \boldsymbol{g}) = \frac{2\sigma_{P,\boldsymbol{f},\boldsymbol{g}} + \gamma}{\sigma_{P,\boldsymbol{f}}^2 + \sigma_{P,\boldsymbol{g}}^2 + \gamma}. \tag{2.33}$$

Combining the three components multiplicatively, the pointwise SSIM yields an error value for each patch P, namely

$$\mathrm{SSIM}(P, \boldsymbol{f}, \boldsymbol{g}) := \ell(P, \boldsymbol{f}, \boldsymbol{g})^{p_1} c(P, \boldsymbol{f}, \boldsymbol{g})^{p_2} s(P\boldsymbol{f}, \boldsymbol{g})^{p_3}. \tag{2.34}$$

Again, p_1, p_2, and p_3 are free parameters that allow exponential weighting of the three components. Assigning the local error values to the centres of each patch yields an error map and averaging over the error map gives the SSIM for the whole image. By construction, the SSIM for any two images is contained in the interval $[-1, 1]$ where a comparison of an image to itself yields the best possible value 1. In contrast to the MSE, lower scores are worse for this quality measure.

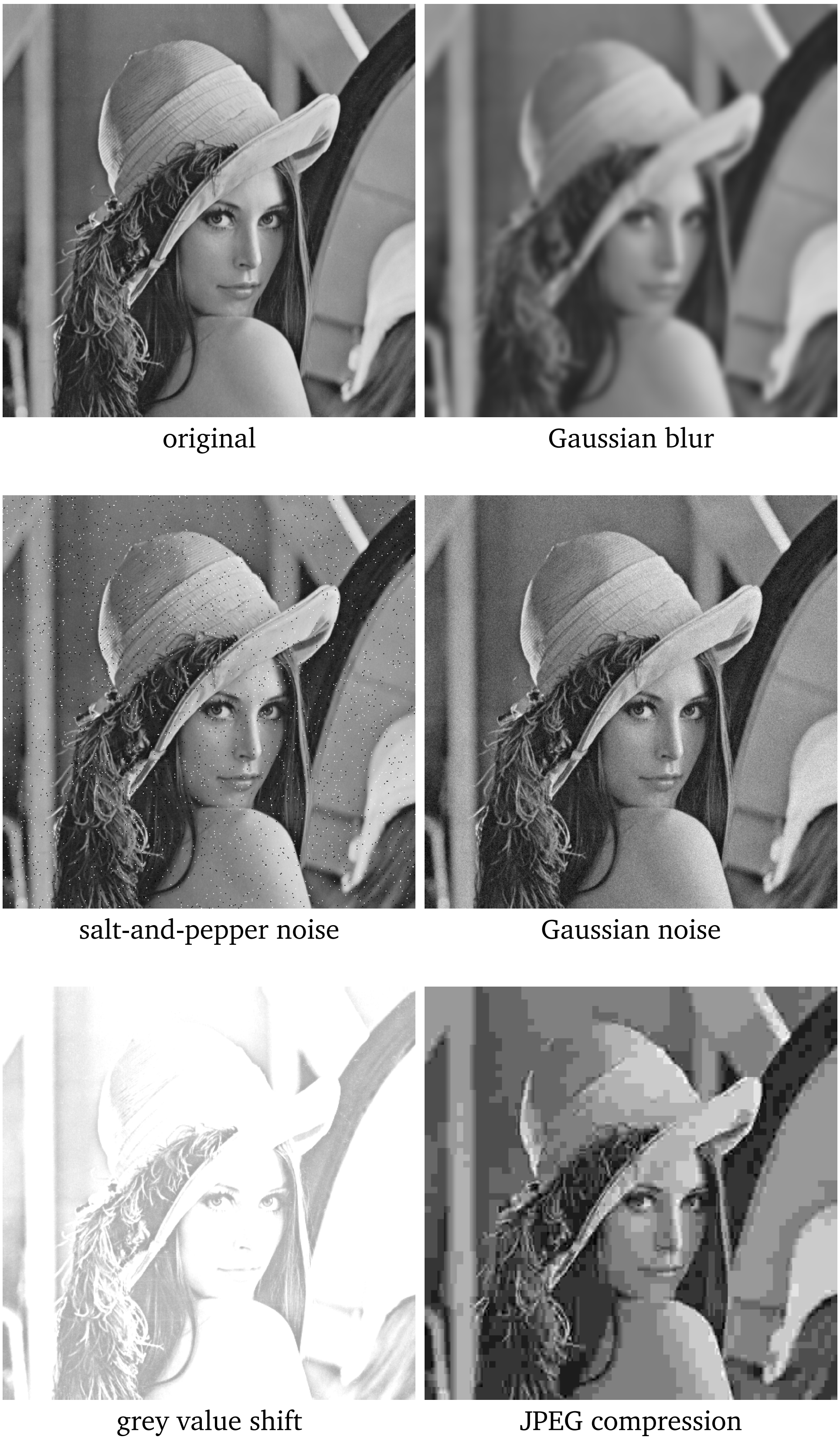
Obviously, the SSIM is significantly more complicated than the MSE and has six free parameters α, β, γ, and p_1, p_2, p_3. This does not count the potential choice for the weights $\boldsymbol{w}$ or the patch sizes. As suggested by Wang et al. [289], we use the standard choices for 8bit images: $p_1 = p_2 = p_3 = 1$, normalised discrete Gaussian weights on an 11×11 window, $\alpha = 6.5025$, $\beta = 58.5225$, and $\gamma = \beta/2$. The parameters α and β depend on the rescaled dynamic range of the image, where the scale is an ad-hoc choice by the authors.

In spite of its relative complexity in comparison to the established quality metrics, the SSIM is becoming increasingly popular in the image processing community. In the wake of its success, a lot of modifications arose: Wang et al. [290] proposed a multi-scale adoption, Sampat et al. [242] defined a wavelet-based structural similarity measure. Both Okarma [210] as well as Kolaman and Yadid-Pecht [153] extend SSIM to colour images, and Zujovic et al. [329] use the ideas of SSIM to compare textures. In addition, Brunet et al. [41] showed that several important properties of the MSE translate to SSIM and its modifications. This is just a small selection from the field, since it is well beyond the scope of this thesis to cover all publications in this direction.

Of course, the SSIM is not the only quality measure that claims to model human perception. There is for example the visual information fidelity (VIF) by Sheikh and Bovik [261] which combines information theory and natural image statistics to establish a connection between perceived image quality and information shared between the reference and the processed image. Overall, the plethora of different metrics creates the problem which metric to use for optimising and evaluating image compression methods. Interestingly, Galić et al. [98] show that in the case of compression, a lot of different metrics like the MSE, SSIM, multi-scale SSIM and VIF lead to similar rankings.

An explanation for this observation is given by Dosselmann and Yang [78]. Not only do they provide additional empirical evidence that MSE and SSIM are correlated, they even propose a conversion algorithm between MSE and SSIM for specific image alterations. While the authors attack the SSIM rather fiercely, their arguments seem to have substance. In particular, we follow their idea to create SSIM hyperspheres and compare them to the behaviour of the MSE.

Figure 2.4 shows the same type of image degradations as in the MSE hypersphere from Figure 2.3, this time with a common SSIM of approximately 0.7, a value which was chosen to give approximately the same amount of Gaussian blur in both hyperspheres. As for the MSE,



original

Gaussian blur

salt-and-pepper noise

Gaussian noise

grey value shift

JPEG compression

Figure 2.4.: **SSIM Hypersphere.** Collection of images with various degradations and an SSIM of approximately 0.7.

the SSIM does not provide the same perceived quality over all images. In fact, the assessment of the grey value shift seems to be even worse in this case.

While the MSE is certainly not a perfect metric, currently no clear standard exists for perceptual metrics. Since the currently most popular perceptual metric, the SSIM, seems to have similar shortcomings as the MSE and the quasi-standards in compression have been optimised in regards to MSE/PSNR, we will also mainly use this simple, quantitative measure for our evaluations.

SSIM Assessment of Colour Images

☞ Our observations in Chapter 8.2 support the results from Wang et al. [289] that the SSIM provides a better assessment of the quality of compressed colour images than the MSE over all colour channels, although the SSIM only works on the brightness information and ignores the colour. This comes from the fact that the human visual system values structure higher than colour [198] and therefore, a pure brightness analysis is more convincing as long as there are no unreasonable colour artefacts. An obvious drawback is that colour does not influence the error at all. Therefore, we always consider a colour-aware error measure in addition.

Related Work

3

> *If I have seen further, it is*
> *by standing on the shoulders of giants.*
>
> — **Sir Isaac Newton**
> (Physicist and Mathematician)

This chapter deals with prior work that forms the foundation of this thesis. Since diffusion is the one main concept that connects all chapters, Section 3.1 offers an in-depth discussion of its physical background and its use in image processing. Of course, diffusion-based image reconstruction is the most important application in the context of compression, but denoising is also discussed due to its connections to Chapter 4 which provides a statistical justification for a large variety of diffusion models. In addition, this chapter also provides a short discussion of numerical implementations.

Section 3.2 recapitulates the history of image compression from the dawn of information theory to the current situation. Thereby, it gives an overview of the field and provides context for the novel contributions of this work. In particular, it focusses on related work in PDE-based compression and the quasi standards in transform-based compression.

Finally, we address the currently most powerful PDE-based general purpose compression method, R-EED by Schmaltz et al. [252]. It forms the basis for some of the compression methods in this thesis and is an important reference for comparisons.

Note that while this chapter covers most of the prior work that is related to this thesis, is not all-encompassing. Since some areas like statistics of natural images in image processing or exemplar-based inpainting are only relevant for individual contributions, a separate discussion of related work is provided in the respective chapters.

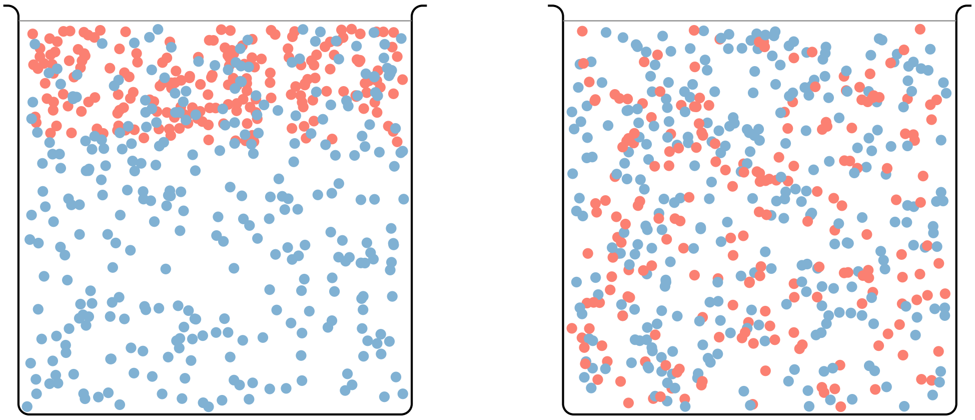


Figure 3.1.: **Diffusion of Iodine.** The figure displays two states of a closed system containing water and iodine. Initially, on the left, the iodine is highly concentrated at the top of the container. Diffusion leads to an equilibration of the concentration: On the right hand side, iodine and water are both uniformly distributed over the whole container.

3.1 Diffusion

> *The differential equations of the propagation of heat express the most general conditions, and reduce the physical questions to problems of pure analysis, and this is the proper object of theory.*
>
> — **Joseph Fourier**
> (Mathematician)

Nowadays diffusion is a popular tool for a wide variety of applications in image processing. However, it originated from a physical model for propagation processes that was thoroughly investigated since the 19th century. Although effects of diffusion were already exploited for practical purposes like metal processing before (see Mehrer and Stolwijk [192]), Graham [109] is considered to be the first to perform a systematic analysis of concentration equilibration in gases and liquids.

A diffusion system considers the concentration of particles at different locations in space. If the distribution is not uniform across the whole system, a transport process that leads to equilibration of the concentration differences takes place. In a classic example by Crank [66], a vessel contains iodine solution and clear water. While there is a distinct separation of the coloured iodine solution and the transparent water in the beginning, over time the whole content of the container assumes a uniform colour (see Figure 3.1).

On the level of individual particles, movement is random according to Brownian motion. Let us consider a so-called isotropic medium, in which movement in all directions is equally probable. At locations where many particles of a given kind are clustered, the probability is higher that at least some of them move in a certain direction. Therefore, the probability that a particle moves from an area of high concentration to an area of low concentration is higher than the other way around. Diffusion in image processing deals with this macroscopic aspect, the flux of particles due to concentration differences.

Following Weickert [293], out of the multitude of different diffusion models we consider those that are described by Fick's Law [92] and correspond to the theories of Fourier [96] on heat transfer. Let $u : \Omega \subset \mathbb{R}^n \times [0, \infty) \to \mathbb{R}$ be a sufficiently often differentiable function that maps a set of spatial coordinates $(x_1, \ldots, x_n)^\top \in \Omega$ at time $t \in [0, \infty)$ to concentrations in a diffusion system. Then, the concentration gradient $\boldsymbol{\nabla} u$ locally describes the direction of the largest concentration difference. According to previous observations on isotropic media, the *flux* $\boldsymbol{j}$ takes place exactly in the inverse direction, i.e. $\boldsymbol{j} = -\boldsymbol{\nabla} u$. Fick's law also incorporates the properties of different media and defines the flux $\boldsymbol{j}$ as

$$\boldsymbol{j} = -\boldsymbol{D}\boldsymbol{\nabla} u. \tag{3.1}$$

Here, the diffusion tensor $\boldsymbol{D}$ is a symmetric and positive definite matrix that describes the propagation properties of a specific system. For example, it accounts for the fact that in crystals, there is a movement bias into specific directions. We discuss the details of its influence on the transport in the next section.

A diffusion model describes only the transport of particles. Thus, mass should be preserved, i.e. no particles are destroyed or created. For any given volume $V \subset \Omega$ with a closed surface ∂V, the change of its concentration over time can be expressed as

$$\partial_t \int_V u \, d\boldsymbol{x} = -\int_{\partial V} \boldsymbol{n}^\top \boldsymbol{j} \, d\boldsymbol{x}. \tag{3.2}$$

This means that the concentration in the volume V can only change over time due to the flux through its surface. Here, $\boldsymbol{n}$ denotes the outwards normal in the corresponding surface point. By exploiting the divergence theorem, one can rewrite Eq. 3.2 as the *continuity equation*

$$\partial_t u = -\text{div}\, \boldsymbol{j}. \tag{3.3}$$

If one considers a closed system such as the container of the iodine example, no flux should occur across the boundaries of the vessel. On $\partial\Omega$ this implies the homogeneous Neumann boundary conditions

$$\boldsymbol{n}^\top \boldsymbol{j} = 0 \quad \Leftrightarrow \quad \boldsymbol{n}^\top \boldsymbol{D}\boldsymbol{\nabla} u = 0 \tag{3.4}$$

where $\boldsymbol{n}$ is again the outward normal in the corresponding boundary point. We can now combine Fick's law from Eq. 3.1 and the continuity equation (3.2) and obtain the *diffusion equation*

$$\partial_t u = \text{div}(\boldsymbol{D}\boldsymbol{\nabla} u). \tag{3.5}$$

This is the fundamental PDE that forms the basis for our compression codecs. In the following, we explain how it can be interpreted as a tool for image processing and what modifications are needed in order to use it for the reconstruction of missing image parts.

3.1.1 From Physics to Image Processing

In image processing, diffusion models consider grey values instead of concentrations. For a given 2-D grey value image $f : \Omega \to \mathbb{R}$, we describe its evolution over time according to diffusion by $u : \Omega \times [0, t) \to \mathbb{R}$. At time $t = 0$, u is initialised with f and grey values are propagated according to the diffusion equation (3.5). We consider a closed system, which means that together with the homogeneous Neumann boundary conditions from Eq. 3.4, the evolution of the image is described by the initial boundary value problem

$$\partial_t u = \operatorname{div}(\boldsymbol{D}\boldsymbol{\nabla} u) \qquad \text{on} \quad \Omega \times [0, \infty), \tag{3.6}$$

$$u(x, y, 0) = f(x, y) \qquad \text{on} \quad \Omega, \tag{3.7}$$

$$\boldsymbol{n}^\top \boldsymbol{D}\boldsymbol{\nabla} u = 0 \qquad \text{on} \quad \partial\Omega \times [0, \infty). \tag{3.8}$$

The physical properties from the previous section reveal a lot about the favourable behaviour of diffusion for image processing tasks such as denoising. If one considers a closed system, Fick's law dictates that mass is neither created, nor destroyed. Interpreting this in terms of images, it means that the *average grey value is preserved*:

$$\mu := \frac{1}{|\Omega|} \int_\Omega f(x, y) \, dx \, dy, \tag{3.9}$$

$$\forall\, t \in [0, \infty) : \quad \frac{1}{|\Omega|} \int_\Omega u(x, y, t) \, dx \, dy = \mu. \tag{3.10}$$

Moreover, we know that the physical diffusion process results in equal concentration at each location. Together with mass preservation, this implies that in the equilibrium, each point contains the average of the original concentrations. This in turn means for image processing, that a diffusion process converges to a *flat steady state* that contains the average grey value in every pixel.

It is even possible to show that during this evolution, the minimum and maximum grey value are never exceeded, i.e. there is a *maximum-minimum-principle*. For practical purposes that implies that neither overshoots nor undershoots occur in image processing with diffusion filters.

$$\forall (x, y, t) \in \Omega \times [0, \infty) : \inf_{(\tilde{x}, \tilde{y}) \in \Omega} f(\tilde{x}, \tilde{y}) \leq u(x, y, t) \leq \sup_{(\tilde{x}, \tilde{y}) \in \Omega} f(\tilde{x}, \tilde{y}) \tag{3.11}$$

For a formal deduction and proofs of these properties in the continuous setting of image processing see Weickert [293]. This book also discusses another important aspect of diffusion-

PDEs: The diffusion time t can be seen as a continuous scale parameter that embeds an original image f in a family of simplified versions $\{T_t f | t \geq 0\}$, fulfilling the semi-group property

$$T_0 f = f, \tag{3.12}$$

$$\forall s, t \geq 0 : T_{t+s} f = T_t(T_s f). \tag{3.13}$$

Weickert [293] provides a detailed overview over different requirements and definitions for scale-spaces, as well as a unifying theory for continuous and discrete diffusion scale-spaces.

Due to the equilibration performed by diffusion one of the most natural applications for diffusion is *denoising*. In the next section, we define different taxonomies for diffusion models and illustrate their properties with denoising examples. The main application for this thesis is however the reconstruction of missing image parts with diffusion which we treat in Section 3.1.3.

Key Idea: Diffusion in Image Processing

Instead of transporting particles and equilibrating concentration, diffusion equilibrates greyvalue differences. The physical properties of diffusion guarantee that image degradations can be removed while the average grey value is preserved and the original range of the image is not exceeded.

3.1.2 Taxonomies and Properties of Diffusion Models

The properties of any diffusion model are determined by its diffusion tensor. Let $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ be a diffusion tensor with eigenvalues $\lambda_1, \ldots, \lambda_n$ and corresponding eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. Then each eigenvalue indicates the amount of diffusion in the direction of the corresponding eigenvector. We follow the taxonomies of Weickert [293] and distinguish diffusion models by *linearity*, *homogeneity*, and *isotropy*.

Linearity: A diffusion tensor $\boldsymbol{D}$ is called *linear*, if it does not depend on the image structure. If it is adapted to the local image structure, i.e. $\boldsymbol{D}(u)$ depends on the value of $u(\boldsymbol{x}, t)$, it is *nonlinear*.

Homogeneity: A *homogeneous* diffusion tensor field has the same value for all spatial coordinates $\boldsymbol{x} \in \Omega$. If it varies with $\boldsymbol{x}$, it is *inhomogeneous*. Meaningful nonlinear diffusion tensors are therefore inhomogeneous. Note that linear diffusion processes can also be inhomogeneous. While this combination of properties is not very frequent in classic image processing, linear inhomogeneous diffusion is a key component of our colour compression codec in Chapter 8.2.

Isotropy: In accordance with the physical definition from the previous section, a diffusion tensor is *isotropic*, if it treats all directions equally. In this case, its eigenvalues are pairwise identical. If it has a directional bias, i.e. at least two eigenvalues differ from each other, it is *anisotropic*.

☞ **Justifying Diffusion**

This chapter motivates diffusion in the classic way: Increasingly complex models evolve from simple ones to improve the quality in concrete applications. An alternative, semantic justification and a unifying probabilistic framework for all filters from this section is given in Chapter 4.

Linear homogeneous diffusion is the simplest possible PDE-model captured by the taxonomies above. Although the linear scale-space concept of Witkin [308] is often cited as the origin for linear diffusion, Weickert et al. [300] have shown that it was first discovered in Japan by Iijima [140, 141]. In the case of linear homogeneous diffusion, the diffusion tensor degenerates to the unit matrix, i.e. $\boldsymbol{D} = \boldsymbol{I}$ and the divergence term of the diffusion equation is nothing else but the Laplace operator:

$$\partial_t u = \Delta u \,. \tag{3.14}$$

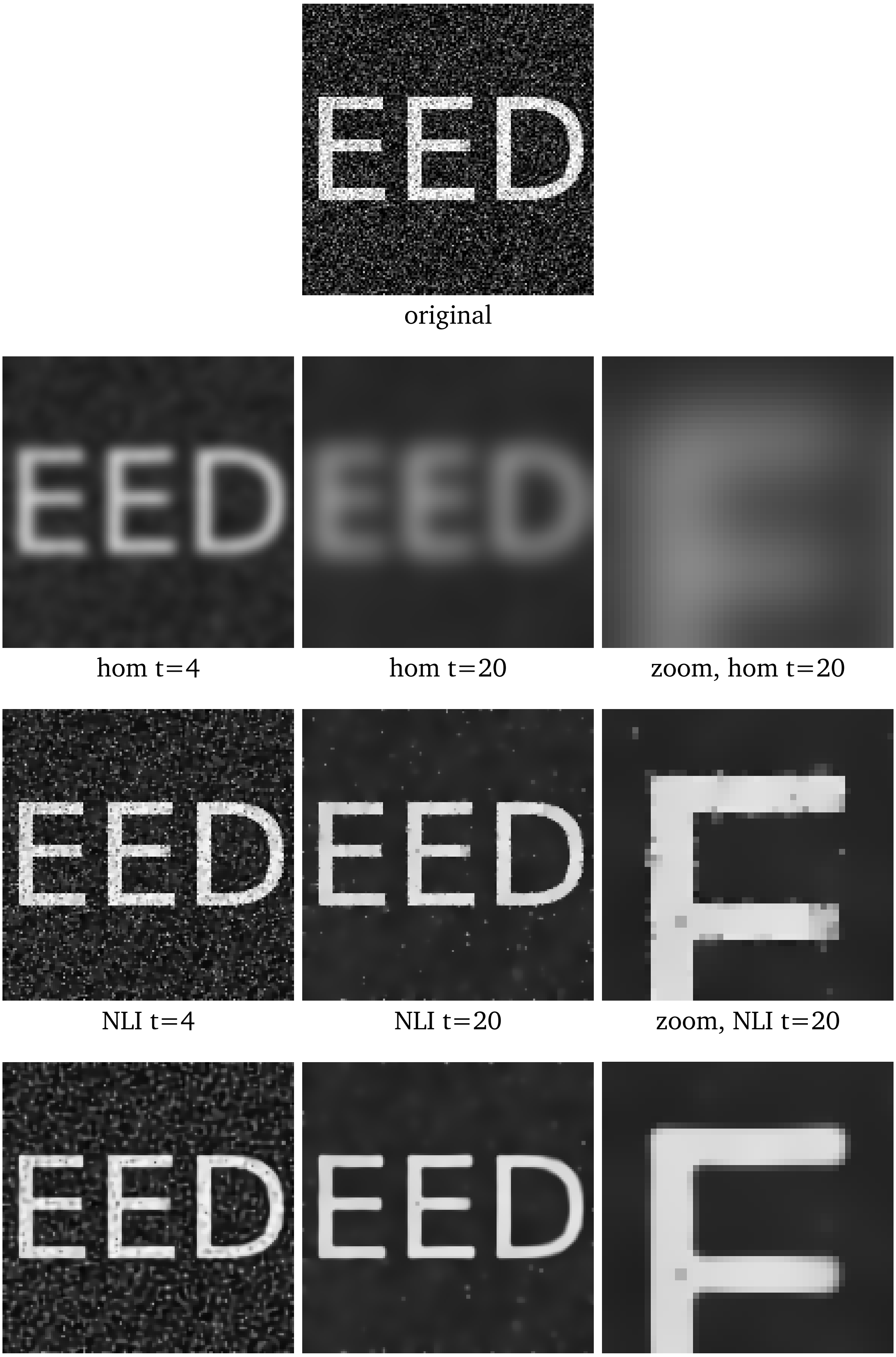
Since this model does not adapt to the local image structure and treats all directions equally, it is able to remove noise from an image, but also blurs and dislocates edges (see Figure 3.2). Linear diffusion is equivalent to Gaussian convolution with a kernel K_σ (see Eq. 2.7), where the variance $\sigma = \sqrt{2t}$ is proportional to the diffusion time t.

A natural remedy for the shortcomings of linear homogeneous diffusion in respect to edges is to identify image discontinuities and stop the diffusion at these locations. Perona and Malik [217] proposed a *nonlinear isotropic diffusion* model that reduces propagation in image regions with large gradient magnitude:

$$\partial_t u = \text{div}(g(|\boldsymbol{\nabla} u|^2)\boldsymbol{\nabla} u) \,. \tag{3.15}$$

Here, the gradient acts as an edge detector. An additional Gaussian presmoothing $u_\sigma := K_\sigma * u$ applied before the gradient computation makes the Perona-Malik filter well-posed and less sensitive to noise (see Catté et al. [50]):

$$\partial_t u = \text{div}(g(|\boldsymbol{\nabla} u_\sigma|^2)\boldsymbol{\nabla} u) \,. \tag{3.16}$$



original

hom t=4

hom t=20

zoom, hom t=20

NLI t=4

NLI t=20

zoom, NLI t=20

EED t=4

EED t=20

zoom, EED t=20

Figure 3.2.: **Denoising with Diffusion.** The images above show diffusion results for homogeneous diffusion (hom), nonlinear isotropic (NLI) and edge-enhancing anisotropic diffusion (EED) at two different diffusion times. NLI and EED both use a Perona Malik diffusivity with the same parameters $\lambda = 5$ and Gaussian presmoothing with $\sigma = 1$.

Since a large gradient magnitude implies a large contrast difference in the image and thereby an edge location, the so-called *diffusivity function* g must be decreasing. Classic diffusivities are continuous, differentiable functions that fulfil the requirements

$$\forall\, x \in [0,\infty) : g(x) > 0, \tag{3.17}$$

$$g(0) = 1, \tag{3.18}$$

$$\forall\, x,y \in [0,\infty) : x \geq y \Rightarrow g(x) \leq g(y). \tag{3.19}$$

Popular representatives for diffusivities that are relevant for this thesis are the Perona/Malik diffusivity g_{PM} [217] and the Charbonnier diffusivity g_{C} [54]:

$$g_{\text{PM}}(s^2) := \left(1 + \frac{s^2}{\lambda^2}\right)^{-1}, \qquad g_{\text{C}}(s^2) := \left(1 + \frac{s^2}{\lambda^2}\right)^{-1/2}. \tag{3.20}$$

In both cases, the *contrast parameter* λ allows to adjust which contrast differences are considered to be edges and should thereby be preserved, and which differences may be removed by the smoothing applied by the diffusion process. This is necessary, since noise itself also constitutes discontinuities.

Note that sometimes, the literature on diffusion in image processing employs other definitions of anisotropy. In particular, nonlinear isotropic diffusion models of the Perona-Malik type are often called anisotropic, even though they treat all directions equally. Figure 3.2 demonstrates that nonlinear isotropic diffusion is indeed able to preserve edges better than linear diffusion. However, it does not solve the problem completely: Either diffusion is inhibited very strongly at discontinuities and thus the noise at such locations remains, or it blurs the edge, leading to the same issues as with linear diffusion.

A representative of truly anisotropic diffusion models according to the taxonomy above is the edge-enhancing anisotropic diffusion (EED) model by Weickert [295]. Not only does it solve the problems of Perona-Malik diffusion, it is also the most important diffusion model for the compression approaches in this thesis. Again, the diffusion is adaptive to the gradient magnitude, but now it uses a diffusion tensor $\boldsymbol{D}$ instead of a scalar diffusivitiy g:

$$\partial_t u = \text{div}(\boldsymbol{D}(|\boldsymbol{\nabla} u_\sigma|^2)\boldsymbol{\nabla} u)\,. \tag{3.21}$$

In order to allow different treatment of individual directions, the diffusion tensor $\boldsymbol{D}$ is chosen in such a way that it adapts to the local structure. This can be achieved by defining the tensor in terms of its eigenvalues and eigenvectors. The first eigenvector $\boldsymbol{v}_1$ is chosen to be parallel to the gradient and thus points across edges. The second eigenvector $\boldsymbol{v}_2$ is perpendicular to $\boldsymbol{v}_1$ and thereby gives the direction along the edge, i.e.

$$\boldsymbol{v}_1 \parallel \boldsymbol{\nabla} u_\sigma, \tag{3.22}$$

$$\boldsymbol{v}_2 \perp \boldsymbol{\nabla} u_\sigma. \tag{3.23}$$

By inhibiting the diffusion in the direction of $\boldsymbol{v}_1$ (across the edge) with a diffusivity g as in Eq. 3.20 and allowing full diffusion along the edge, one obtains

$$\boldsymbol{D}(|\boldsymbol{\nabla} u_\sigma|^2)) := g(|\boldsymbol{\nabla} u_\sigma|^2) \cdot \boldsymbol{v}_1 \boldsymbol{v}_1^\top + 1 \cdot \boldsymbol{v}_2 \boldsymbol{v}_2^\top. \tag{3.24}$$

Considering again the comparison in Figure 3.2: EED preserves the sharp edges of the text and also removes noise at these locations. The difference to isotropic diffusion becomes visible by zooming in.

More Diffusion Models

The three models discussed here are by far not the only diffusion models. In Chapter 4, we introduce and discuss a larger variety of diffusion PDEs and their relation to variational methods and statistical image priors.

3.1.3 Diffusion Inpainting

Inpainting is an application in image processing that has originated from image restoration [26]. If parts of images are missing or damaged (e.g. scratches on old photographs as in Figure 3.3), inpainting has the goal to reconstruct these regions. One could also speak of the interpolation of missing data from the known image. Of course, interpolation is a vast field that spans a lot of different approaches like spline interpolation. We only cover closely related methods in this thesis. For a comprehensive overview over the large variety of alternatives we refer the reader to Meijering [193]. Besides the application of inpainting in restoration and compression, inpainting can also be used for image upsampling [9, 21, 24, 185].

Hoeltgen et al. [125] have pointed out in their review on PDE-based compression that the idea of filling in missing areas with diffusion has been implicitly used for a long time in optic flow [130] before PDE-based inpainting became popular. During the computation of a displacement vector field on image sequences, a variational approach related to the diffusion PDEs from the previous sections complements the missing parts of the field.

The early works of Masnou and Morel [190] considered a variational model for inpainting based on level-lines as a disocclusion problem. The goal was here to reconstruct the missing or occluded parts according to human perception that is described by Kanizsa [148] and Werner

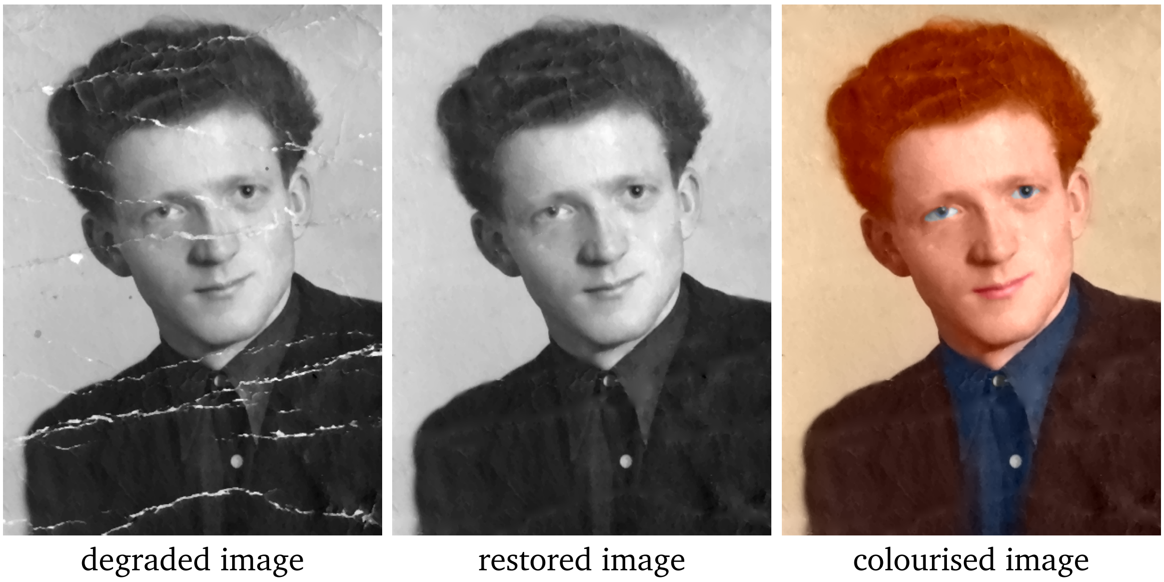


Figure 3.3.: **Image Restoration with PDE-based Inpainting.** By specifying the scratches on an old photograph as missing image regions, inpainting can be used for image restoration. **Left:** Authentic photograph of the author's grandfather with heavy degradations. **Middle:** Restoration by EED inpainting. The locations of the scratches were manually selected. **Right:** Colourisation with EED inpainting according to the novel technique presented in Chapter 8.2.

[307]. Indeed, this theory describes the strengths of PDE-based inpainting well and motivates the experiments in Figure 3.4.

After Bertalmío et al. [26] used PDEs to mimic the procedures conducted by human conservators, many other approaches in image inpainting followed [16, 48, 51, 115, 281, 302]. Some of them go beyond the original goals of 2-D image inpainting and work instead on vector [17] or tensor fields [297].

A very general formulation of the PDE-based inpainting problem is given by Weickert and Welk [297]. Let L denote a suitable differential operator, e.g. one of the diffusion operators from Section 3.1. The goal is now to compute a reconstruction u of a given image f. This reconstruction can be obtained as the solution of the *general inpainting equation*

$$(1 - c(\boldsymbol{x}))Lu - c(\boldsymbol{x})(u - f) = 0. \tag{3.25}$$

Here, the difference $u - f$ models the deviation of the reconstruction u from the original f and thereby defines a criterion for *approximation*. The actual filling-in is performed by an application Lu of the differential operator to the reconstructed image u. It specifies a smoothness constraint on the solution u. Both terms are weighted by the *confidence function* $c(\boldsymbol{x}) : \Omega \to [0, 1]$. For $c(\boldsymbol{x}) = 0$, data is purely interpolated due to the constraints imposed by the differential operator, for $c(\boldsymbol{x}) = 1$ the reconstruction is forced to perfectly match the original. All values in-between give a continuous confidence rating of the given data: The higher the amount of suspected degradation at the point $f(\boldsymbol{x})$, the smaller $c(\boldsymbol{x})$ should be.

For our purposes in image compression, we only consider binary functions c. In this case, we perform pure *interpolation*, since the known data is preserved exactly. We assume that the image domain Ω is partitioned into a known image part, the *inpainting mask* K and an unknown image part, the *inpainting domain* $\Omega \setminus K$. For the inpainting mask, we require that the known data is perfectly reconstructed, i.e.

$$\forall \boldsymbol{x} \in K : u(\boldsymbol{x}) = f(\boldsymbol{x}). \tag{3.26}$$

On the other hand, the confidence in the missing data is assumed to be zero. Therefore, $c(\boldsymbol{x})$ can be expressed by the indicator function χ_K of the inpainting mask K:

$$c(\boldsymbol{x}) = \chi_K(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in K, \\ 0 & \text{else.} \end{cases} \tag{3.27}$$

This allows to simplify Eq. 3.25 significantly, since the partition into inpainting mask and domain allows to split it into the two equations

$$\forall \boldsymbol{x} \in K : \quad u(\boldsymbol{x}) - f(\boldsymbol{x}) = 0, \tag{3.28}$$

$$\forall \boldsymbol{x} \in \Omega \setminus K : \quad Lu(\boldsymbol{x}) = 0 \tag{3.29}$$

If one considers the inpainting operator L to be the divergence term of the general diffusion model from Eq. 3.5, Eq. 3.29 requires that u is a steady state of a diffusion process. The difference to denoising is the specification of fixed known data from Eq. 3.28. As for denoising, we can specify the evolution of the solution u as an initial boundary value problem, where Eq. 3.28 imposes Dirichlet boundary conditions to Eq. 3.6:

$$\partial_t u = \operatorname{div}(\boldsymbol{D}\boldsymbol{\nabla} u) \qquad \text{on} \quad \Omega \setminus K \times [0, \infty), \tag{3.30}$$

$$u(x, y, t) = f(x, y) \qquad \text{on} \quad K \times [0, \infty), \tag{3.31}$$

$$\boldsymbol{n}^\top \boldsymbol{D}\boldsymbol{\nabla} u = 0 \qquad \text{on} \quad \partial\Omega \times [0, \infty). \tag{3.32}$$

It has been either shown or empirically verified for all diffusion models previously discussed that the initialisation at $t = 0$ on $\Omega \setminus K$ does not influence the steady state. However, the initialisation still has an impact on practical applications: In Chapter 7, we discuss its effect on convergence speed and show how to exploit this connection to obtain real-time performance in video decoding.

A comparison of the three most important representatives of inpainting PDEs in Figure 3.4 shows that EED yields by far the best results. Linear diffusion propagates information in all directions equally and fails to reconstruct any sharp image edges. While nonlinear isotropic diffusion is indeed able to inhibit the propagation very strongly, it suffers from similar problems as in the case of denoising. In order to propagate an edge into the inpainting domain, diffusion needs to be limited to the direction along the edge, which is not possible if the process is isotropic. However, EED can also reconstruct sharp edges in the missing data.

The suitability of different inpainting operators for compression has already been investigated on several occasions. Multiple publications that deal with inpainting from random or optimally chosen scattered data [34, 59, 99, 125] have confirmed that EED produces superior results in comparison to linear homogeneous diffusion and other operators such as biharmonic and triharmonic inpainting [81] or total variation (TV) [240] interpolation.

In the context of compression, Galić et al. [100] have shown that EED outperforms linear homogeneous diffusion at the same compression ratio, if known data is selected and stored with a triangular subdivision scheme. A more extensive comparative study with rectangular subdivision by Schmaltz et al. [252] includes also the absolute minimal Lipschitz extension (AMLE) [11] and the method of Bornemann and März [30]. The latter is an extension of the single-pass algorithm of Telea [278]. Moreover biharmonic, triharmonic, and TV inpainting are also considered in this comparison. Again, EED produced the best compression results. However, it should be noted that both methods use similar selection and storage strategies for the known data. In Chapter 5.3.1 we investigate how far this influences the performance of PDE-based inpainting.

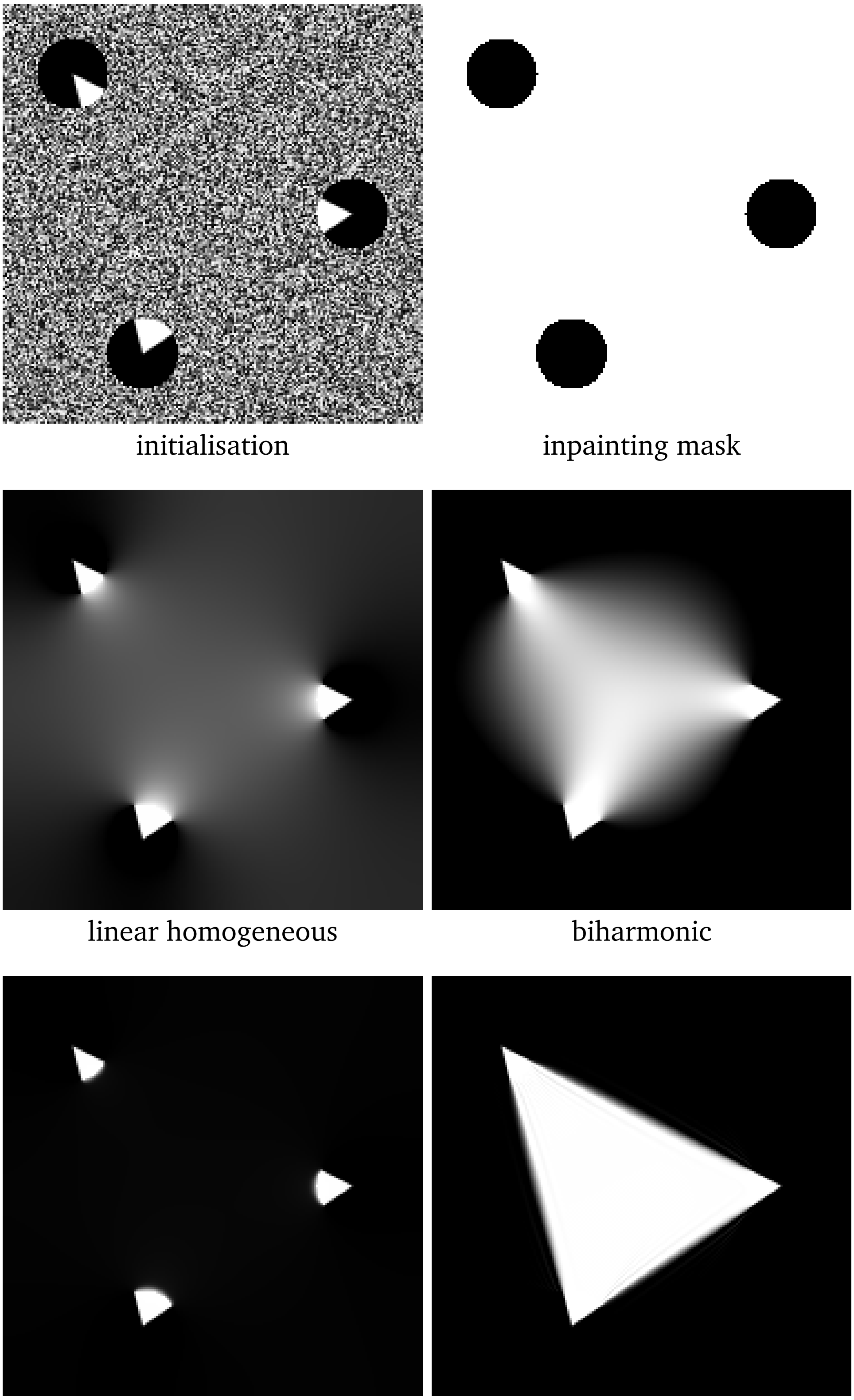
3.1.4 Discretisations and Solvers

The previous sections on diffusion cover purely continuous models. For the purpose of practical compression codecs with diffusion-based inpainting, it is necessary to discretise the aforementioned models and to solve the resulting discrete inpainting equation. While we do not deal with the development or in-depth analysis of discretisations or solvers in this thesis, it is nevertheless important to discuss them.

Finite Difference Approximations

All discrete implementations of diffusion models in this thesis rely on the approximation of their derivatives by finite differences. The general concept can be explained with the example of a 1-D function $u(x) : \Omega \subset \mathbb{R} \to \mathbb{R}$. For example, the derivative at a point $u(x)$ on the discrete grid with grid size h can be approximated by the difference between the neighbouring pixels $u_{i-1} := u(x-h)$ and $u_{i+1} := u(x+h)$, the so-called *central difference*

$$u_i' \approx \frac{u_{i+1} - u_{i-1}}{2h}. \tag{3.33}$$



initialisation

inpainting mask

linear homogeneous

biharmonic

nonlinear isotropic

nonlinear anisotropic (EED)

Figure 3.4.: **Inpainting with Diffusion.** A triangle should be reconstructed from the known data marked in black in the inpainting mask. EED is able to reproduce sharp image edges, while linear operators lead to blurry structures. Nonlinear isotropic diffusion either produces similar results to the homogeneous methods or inhibits the propagation very strongly, depending on the choice of the contrast parameter.

Assuming that u is sufficiently often differentiable, it can be approximated by a Taylor series at each point in Ω. This comes down to

$$u(x+h) = \sum_{k=0}^{n} \frac{h^k}{k!} f^{(k)}(x) + \mathcal{O}(h^{n+1}), \tag{3.34}$$

$$f \in \mathcal{O}(g) :\Leftrightarrow 0 \leq \limsup_{x \to a} \frac{|f(x)|}{|g(x)|} < \infty, \quad a \in \mathbb{R} \cup \{-\infty, \infty\}, \tag{3.35}$$

where $\mathcal{O}$ is the *Bachmann-Landau notation*. It describes the asymptotic behaviour of f. If the absolute value of f does not grow faster than the function g, we have $f \in \mathcal{O}(g)$. If one computes Taylor approximations of the two points used to compute the central difference

$$u(x-h) = u_{i-1} = u_i - hu_i' + \frac{h^2}{2} u_i'' - \frac{h^3}{6} u_i''' + \frac{h^4}{24} u_i'''' + \mathcal{O}(h^5), \tag{3.36}$$

$$u(x+h) = u_{i+1} = u_i + hu_i' + \frac{h^2}{2} u_i'' + \frac{h^3}{6} u_i''' + \frac{h^4}{24} u_i'''' + \mathcal{O}(h^5) \tag{3.37}$$

and plugs them into the finite difference Eq. 3.33, one obtains the error term

$$\frac{u_{i+1} - u_{i-1}}{2h} = \underbrace{\frac{1}{2h}(1-1)}_{=0} u + \underbrace{\frac{1}{2}(1+1)}_{=1} u_i' + \underbrace{\frac{h}{4}(1-1)}_{=0} u_i'' + \underbrace{\frac{h^2}{12}(1+1)}_{\neq 0} u_i''' + \mathcal{O}(h^3) \tag{3.38}$$

$$= u_i' + \mathcal{O}(h^2). \tag{3.39}$$

The error term with the smallest power of h, the so-called order of consistency, determines the quality of the approximation for $h \to 0$. In a similar way, higher order derivatives can also be expressed by finite differences. Applying this principle to the spatial derivatives contained in the divergence term of the diffusion equation (3.6) is a straightforward way to obtain the *standard discretisation* [293].

Although it is easy to discretise a diffusion PDE with a reasonable order of consistency, the filter results are still influenced significantly by the choice of discretisation. Weickert et al. [299] have proposed a unifying framework for a large number of non-standard finite difference discretisations for anisotropic diffusion [65, 293, 294, 304, 305]. They use the following finite differences as building blocks for the discretisation on a (2×2)-pixel neighbourhood $\{i,\, i{+}1\} \times \{j,\, j{+}1\}$ around a pixel $(i + \frac{1}{2}, j + \frac{1}{2})$ on a staggered grid:

$$\boxed{\underrightarrow{\quad}} := D_x^- u_{i+\frac{1}{2}, j+\frac{1}{2}} := \frac{u_{i+1,j} - u_{i,j}}{h_x}, \tag{3.40}$$

$$\boxed{\overrightarrow{\quad}} := D_x^+ u_{i+\frac{1}{2}, j+\frac{1}{2}} := \frac{u_{i+1,j+1} - u_{i,j+1}}{h_x}, \tag{3.41}$$

$$\boxed{\vert\;\;} \;:=\; D_y^- u_{i+\frac{1}{2},j+\frac{1}{2}} \;:=\; \frac{u_{i,j+1}-u_{i,j}}{h_y}, \tag{3.42}$$

$$\boxed{\;\;\vert} \;:=\; D_y^+ u_{i+\frac{1}{2},j+\frac{1}{2}} \;:=\; \frac{u_{i+1,j+1}-u_{i+1,j}}{h_y}. \tag{3.43}$$

Here, h_x and h_y denote the respective grid sizes in the x- and y-direction. Weickert et al. [299] propose the following discretisations for the derivative terms occurring in the diffusion equation in pixel $(i+\frac{1}{2},\, j+\frac{1}{2})$:

$$J_{1,1} = (\partial_x u_k)^2 \approx (1-\alpha)\frac{1}{2}\left(\boxed{\overline{\;\;}}^{\,2} + \boxed{\underline{\;\;}}^{\,2}\right) + \alpha\cdot\left(\boxed{\overline{\;\;}}\cdot\boxed{\underline{\;\;}}\right), \tag{3.44}$$

$$J_{2,2} = (\partial_y u_k)^2 \approx (1-\alpha)\frac{1}{2}\left(\boxed{\;\;\vert}^{\,2} + \boxed{\vert\;\;}^{\,2}\right) + \alpha\cdot\left(\boxed{\;\;\vert}\cdot\boxed{\vert\;\;}\right), \tag{3.45}$$

$$J_{1,2} = \partial_x u_k\,\partial_y u_k \approx (1-\beta)\frac{1}{4}\left(\boxed{\underline{\;\;}}\boxed{\vert\;\;} + \boxed{\overline{\;\;}}\boxed{\;\;\vert}\right) + (1+\beta)\frac{1}{4}\left(\boxed{\underline{\;\;}}\boxed{\;\;\vert} + \boxed{\overline{\;\;}}\boxed{\vert\;\;}\right) \tag{3.46}$$

with free parameters $\alpha \in [0, \frac{1}{2}]$ and β with $|\beta| \leq 1-2\alpha$. These derivative terms coincide with components of the structure tensor $\boldsymbol{J}$ discussed later in Chapter 4. The restrictions on these parameters stem from stability considerations. Note that in the most general version of the framework, the parameters are actually spatially variant, i.e. the discretisation in each grid point can have different α and β. In particular, for a given diffusion tensor of the form

$$\boldsymbol{D} = \begin{pmatrix} a & b \\ b & c \end{pmatrix}, \tag{3.47}$$

the sign of β is locally adapted to the sign of the component b in order to reduce potential over- or undershoots. For convenience of notation, β is in practice often substituted by a global parameter $\gamma \in [-1, 1]$ with $\beta := \gamma(1-2\alpha)\mathrm{sign}(b)$. For more details on these issues we refer to Weickert et al. [299].

The experiments of Weickert et al. [299] show that in concrete applications like demosaicking, the standard discretisation produces a significantly higher error (≈ 10dB in PSNR) than the best selection of parameters for the aforementioned discretisation. We also provide some additional evidence for the potential of this discretisation in Chapter 8 in the context of image colourisation. Also, in the case of directional image statistics, were rotational invariance is important, we resort to a suitable special case of the framework above, namely the LSAS scheme [305] with $\alpha \in [0, \frac{1}{2}]$ and $\beta = 0$.

However, we should note that choosing optimal discretisations is application dependent and still an unsolved problem. We illustrate this problem with some experiments in Chapter 8.2.2. Furthermore, we want to eliminate the discretisation as a confounding factor in our evaluation. If we compare one of our new compression schemes to e.g. the R-EED method, which uses a standard discretisation, we want to have equal conditions for both methods.

No matter which discretisation we choose, the approximation of spatial derivatives by finite differences yields a semi-implicit discretisation of the form

$$\frac{d}{dt}\boldsymbol{u} = \boldsymbol{A}(\boldsymbol{u})\boldsymbol{u}, \tag{3.48}$$

where $\boldsymbol{u} \in \mathbb{R}^{n_x n_y}$ is a discrete image with resolution $n_x \times n_y$ that still depends on a continuous diffusion time t and $\boldsymbol{A} \in \mathbb{R}^{n_x n_y \times n_x n_y}$ is the finite difference approximation of the divergence term. With the discretisation by Weickert et al. [299] discussed above and the notation from Eq. 3.47 for the entries a, b, and c of the diffusion tensor, each line of the matrix $\boldsymbol{A}(\boldsymbol{u})$ corresponds to the following *stencil*:

$$\frac{1}{2h^2} \cdot \begin{array}{|c|c|c|} \hline \Big[(\beta-1)b+\alpha(a+c)\Big]_{i-\frac{1}{2},j+\frac{1}{2}} & \begin{array}{l} \Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i+\frac{1}{2},j+\frac{1}{2}} \\ +\Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i-\frac{1}{2},j+\frac{1}{2}} \end{array} & \Big[(\beta-1)b+\alpha(a+c)\Big]_{i+\frac{1}{2},j+\frac{1}{2}} \\ \hline \begin{array}{l} \Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i-\frac{1}{2},j+\frac{1}{2}} \\ +\Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i-\frac{1}{2},j-\frac{1}{2}} \end{array} & \begin{array}{l} -\Big[(1-\alpha)(a+c)-(\beta-1)b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \\ -\Big[(1-\alpha)(a+c)-(\beta-1)b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \\ -\Big[(1-\alpha)(a+c)-(\beta-1)b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \\ -\Big[(1-\alpha)(a+c)-(\beta-1)b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \end{array} & \begin{array}{l} \Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i+\frac{1}{2},j+\frac{1}{2}} \\ +\Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \end{array} \\ \hline \Big[(\beta-1)b+\alpha(a+c)\Big]_{i-\frac{1}{2},j-\frac{1}{2}} & \begin{array}{l} \Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \\ +\Big[(1-\alpha)c+\alpha a-\beta b\Big]_{i-\frac{1}{2},j-\frac{1}{2}} \end{array} & \Big[(\beta-1)b+\alpha(a+c)\Big]_{i+\frac{1}{2},j-\frac{1}{2}} \\ \hline \end{array} \tag{3.49}$$

The stencils contains the weights corresponding to the pixel i, j (the centre of the stencil) and its 3×3 neighbours. We have assumed here an equal grid size $h := h_x = h_y$ for both directions. Throughout the whole thesis, we use this general stencil, only the choice of α and β varies between the different applications.

In the following we discuss two different ways to obtain a numerical solution of the diffusion equation by discretising the time derivative in different ways.

Explicit Scheme

First, we consider a discretisation of the time derivative in Eq. 3.48 with the forward difference

$$\frac{d}{dt}\boldsymbol{u} \approx \frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^{k}}{\tau}, \tag{3.50}$$

$$\boldsymbol{u}^{k} := \boldsymbol{u}(k\tau). \tag{3.51}$$

Here, τ is the temporal grid size or *time step size*. This discretisation leads to the *explicit* scheme

$$\frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^{k}}{\tau} = \boldsymbol{A}(\boldsymbol{u}^{k})\boldsymbol{u}^{k}\,. \tag{3.52}$$

Computing the evolving image $\boldsymbol{u}$ for diffusion times that are a multiple of the time step size is very straightforward here, since after a small rearrangement of Eq. 3.52, the next iterate $\boldsymbol{u}^{k+1}$ can be computed from $\boldsymbol{u}^{k}$ with the simple matrix-vector multiplication

$$\boldsymbol{u}^{k+1} = (\boldsymbol{I} + \tau\boldsymbol{A}(\boldsymbol{u}^{k}))\boldsymbol{u}^{k}. \tag{3.53}$$

Unfortunately, explicit schemes suffer from limitations of the time step size τ due to stability reasons [293, 299]. In particular for inpainting, where large diffusion times can be necessary to reach the steady state of the diffusion process, this is a significant drawback.

The fast explicit diffusion (FED) scheme by Weickert et al. [114, 298] provides a remedy for this issue. It replaces the fixed time step size τ with cycles of varying step size. Individual iterations in each cycle can exceed the stability limit significantly, but each cycle itself is stable. Since FED is fast, well-suited for GPU (graphics processing unit) implementations, and a reference implementation is freely available online [113], this is the primary solver that we use for our implementations in this thesis.

Semi-Implicit Scheme

Another possibility to circumvent the time step limitations of explicit schemes is a different discretisation of the time derivative that leads to the *semi-implicit* discretisation of Eq. 3.48. These schemes were popular in PDE-based compression with anisotropic diffusion before the advent of FED. A backward difference in time leads to

$$\frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^{k}}{\tau} = \boldsymbol{A}(\boldsymbol{u}^{k})\boldsymbol{u}^{k+1}, \tag{3.54}$$

$$\Leftrightarrow \quad (\boldsymbol{I} - \tau\boldsymbol{A}(\boldsymbol{u}^{k}))\boldsymbol{u}^{k+1} = \boldsymbol{u}^{k}. \tag{3.55}$$

Although the resulting scheme seems to be similar to the explicit method, there are some key differences. It can be shown that τ can be chosen arbitrarily large [293], but this advantage comes at a cost. Instead of iterating a simple matrix-vector multiplication, semi-implicit schemes require solving a linear system of equations with the system matrix $(\boldsymbol{I} - \tau \boldsymbol{A}(\boldsymbol{u}^k))$.

Two solvers that are relevant for this thesis both rely on the family of Jacobi methods. In the following we briefly discuss these solvers, loosely following Barrett et al. [18, Chapter 2]. Jacobi methods solve a system of equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\boldsymbol{x}$, $\boldsymbol{b} \in \mathbb{R}^n$ iteratively. Starting from a value $\boldsymbol{x}^0$, they compute a recursively defined series $(\boldsymbol{x}^k)_{k \in \mathbb{N}}$ that converges to the solution $\boldsymbol{x}$.

The original *Jacobi method* considers each equation of the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ individually. For every equation of the form $(\boldsymbol{A}\boldsymbol{x})_i = b_i$, x_j with $j \neq i$ are kept constant. This allows to solve the equation for x_i:

$$\sum_{j=1}^{n} a_{i,j} x_j = b_i, \tag{3.56}$$

$$\Leftrightarrow \quad x_i = \frac{b_i - \sum_{j \neq i} a_{i,j} x_j}{a_{i,i}}. \tag{3.57}$$

This leads to the iterative scheme

$$x_i^k = \frac{b_i}{a_{i,i}} - \sum_{j \neq i} \frac{a_{i,j}}{a_{i,i}} x_j^{k-1}. \tag{3.58}$$

For such an iterative scheme, it is unclear how close the iterate $\boldsymbol{x}^k$ is to the actual solution $\boldsymbol{x}$, since the error $\boldsymbol{x} - \boldsymbol{x}^k$ cannot be computed. However, it is possible to determine the *residual* [194], which describes the effect of the error on the right hand side of the system of equations:

$$\boldsymbol{r}^k := \boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}^k) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\boldsymbol{x}^k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^k. \tag{3.59}$$

In many cases during this thesis, we use the relative residual $|\boldsymbol{r}^0|/|\boldsymbol{r}^k|$ to determine when to terminate the iteration. This criterion can also be used for explicit FED schemes, since the residual of the semi-implicit scheme can also be computed for any FED iterate. Considering again the Jacobi method in Eq. 3.58, one can also describe it in terms of the residual:

$$x_i^k = \frac{b_i}{a_{i,i}} - \sum_{j \neq i} \frac{a_{i,j}}{a_{i,i}} x_j^{k-1} = \frac{b_i}{a_{i,i}} + \frac{a_{i,i} x_i^k - (\boldsymbol{A}\boldsymbol{x}^k)_i}{a_{i,i}} \tag{3.60}$$

$$= x_i^k + \frac{(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^k)_i}{a_{i,i}} = x_i^k - \underbrace{\frac{r_i^k}{a_{i,i}}}_{\text{correction}} \tag{3.61}$$

Thus, the Jacobi method can be interpreted as a series of correction steps according to the rescaled residual.

Gauß-Seidel solvers extend the Jacobi method by exploiting the fact that for a sequential computation of $\boldsymbol{x}_1^k, \dots, \boldsymbol{x}_n^k$, the results for $\boldsymbol{x}_j^k$ with $j < i$ are already available when $\boldsymbol{x}_i^k$ is computed. Supplementing these results into Eq. 3.58 yields

$$x_i^k = \frac{b_i}{a_{i,i}} - \left(\sum_{j<i} \frac{a_{i,j}}{a_{i,i}} x_j^k + \sum_{j>i} \frac{a_{i,j}}{a_{i,i}} x_j^{k-1} \right). \tag{3.62}$$

Incorporating the already computed results into the following computation speeds up convergence in general.

With the Gauß-Seidel solver, we can finally explain the method that the reference implementation of R-EED [252] uses (see Section 3.3). *Successive overrelaxation (SOR)* computes a weighted average of the Gauß-Seidel iterates x_i^k and x_i^{k-1}:

$$x_i^k = \omega \left(\frac{b_i}{a_{i,i}} - \left(\sum_{j<i} \frac{a_{i,j}}{a_{i,i}} x_j^k + \sum_{j>i} \frac{a_{i,j}}{a_{i,i}} x_j^{k-1} \right) \right) + (1-\omega) x_i^{k-1} \tag{3.63}$$

In fact, for $\omega = 1$ SOR and Gauß-Seidel are equivalent. However, if one chooses $\omega < 1$, the Gauß-Seidel correction is attenuated and for $\omega > 1$, it is increased. Such an *overrelaxation* can improve the speed of convergence. For symmetric, positive definite matrices, SOR converges for $\omega \in (0, 2)$. Outside of this interval, no convergence can be obtained in general [287].

Although SOR is considerably faster than the original Jacobi method and was therefore popular in PDE-based compression, it also has some drawbacks. On one hand, the SOR steps depend on the order in which the image pixels are considered. On the other hand, it is strictly sequential and thereby not suitable for CPU or GPU parallelisation.

However, there is an alternative to SOR that relies on the same ideas as the FED scheme. It has been shown by Grewenig et al. [298] that Fast-Jacobi (FJ), a cyclic extension of the Jacobi overrelaxation scheme [287], is particularly useful for diffusion problems with strongly varying coefficients. It also preserves the positive properties of FED, namely that it can be parallelised in a straightforward way.

Fast Inpainting with PDEs

The discretisations discussed in this section do not consider inpainting yet, but only model a diffusion process for image evolution as it is applied e.g. in denoising. However, the discretisation itself remains the same, it is just embedded into the diffusion model in Eq. 3.30 – Eq. 3.32. Chapter 7 compares different solvers for inpainting in a video compression context and introduces additional means for speed ups such as parallelisation and coarse-to-fine strategies.

3.2 A Short History of Image Compression

> *Those who cannot remember the past*
> *are condemned to repeat it.*
>
> — **George Santayana**
> (Philosopher)

A revolutionary seminal paper by Claude E. Shannon [258, 259] laid the foundation for image compression in 1948. It single-handedly established a whole new area of research: *information theory*. At its core, it deals with communication in a very broad sense: A transmitter encodes a message and transports it to a receiver via a channel. Such a model includes the case where the sender is a compression program that has the goal to transfer an efficient representation of an image to a storage device. In his foreword to the book version of Shannon's original paper, Warren Weaver even mentions explicitly, that Shannon's definition of communication includes "the pictorial arts" [260, p. 3]. The original idea of transmitting pictures electronically dates back even further. Alexander Bain registered a patent for a fax machine already in 1843 [15]. However, it took a long time until a corresponding mathematical theory was developed.

Beyond the general concept of encoding and transmitting data efficiently, the most direct effect of Shannon's work on image compression is the concept of entropy and its implications for coding. In the following, we first discuss these general purpose compression algorithms that only remove *redundancies* in the representation of data and allow to recover the original input again exactly. These *lossless* encoders are also an important building block for *lossy* image compression codecs which discard visually unimportant information (*irrelevancies*) to achieve smaller file sizes (see e.g. [268]).

3.2.1 Entropy Coding

The general goal of lossless compression is to represent a given message or *source word* with the smallest possible amount of symbols such that the original message can still be perfectly reconstructed. The message is composed from symbols from a *source alphabet* $S = \{s_1, \ldots, s_m\}$ and should be encoded by a sequence of symbols from the *code alphabet* $A = \{a_1, \ldots, a_n\}$. For the purpose of image compression, $S = \{0, \ldots, 255\}$ could for example denote the grey values of an image that should be represented by a binary code from $A = \{0, 1\}$.

In information theory, the source is modelled statistically. Each symbol s_i occurs with a probability $p_i \in [0, 1]$ in the source word. These probabilities are usually estimated from the source file or follow an underlying model for the message content (e.g. the occurrence frequencies of letters in the English language or the statistics of a certain class of images). First, we consider a simple model that assumes that each symbol in the source word is randomly selected from the alphabet and independent of the other symbols.

In such a compression model, the probability of a symbol defines how much information it carries. If one only considers symbols that actually occur, i.e. have a probability greater than zero, their information content should be specified by a non-negative number. Intuitively, if the probability of a symbol is 1, it contains no information, since all messages composed from this symbol are trivial. Thus, self-information should be zero exactly for this case. These three requirements lead to an axiomatic definition of the *self-information* $I(s_i)$ according to

$$\forall p_i \in (0,1]: \quad I(s_i) \geq 0, \tag{3.64}$$

$$\forall p_i \in (0,1): \quad I(s_i) > 0, \tag{3.65}$$

$$\forall p_i, p_j \in (0,1): \quad I(s_i s_j) = I(s_i) + I(s_j). \tag{3.66}$$

Aczél and Daróczy [5] have shown that these axioms enforce a logarithmic relation between self-information and probability of occurrence. Thereby, we define $I(s_i)$ as

$$I(s_i) = \log_2 \frac{1}{p_i} = -\log_2 p_i. \tag{3.67}$$

The higher the amount of self-information, the more expensive a symbol is to encode. In particular, the $I(s_i)$ specifies the amount of bits that is needed to encode this particular symbol. For a whole code-word, the expected value of the average self-information of the whole source alphabet is called the *entropy*

$$H(S) := E(I(S)) = \sum_{i=1}^{m} p_i I(s_i) = -\sum_{i=1}^{m} p_i \log_2 p_i. \tag{3.68}$$

It can be shown that this entropy provides a lower bound for the average length of the code words produced by any compression scheme where the output can be deterministically decoded again. Going into more detail here is beyond the scope of this thesis. Therefore we refer to the mathematically compelling discussion of the topic provided by Hankerson et al. [119].

Algorithms that have the goal to come as close to the aforementioned theoretical limit as possible are called *entropy coders*. Probably the first binary entropy encoder was *Morse code* [285] which was developed for practical purposes before an overarching theory for message coding existed. However, it is far from optimal.

Other algorithms like Shannon coding [258] or Shannon-Fano coding [91, 258] provide better results, but are not used in practice since Huffman [136] proposed a simple algorithm in 1952 that offers a good combination of high compression ratios and fast performance. It orders symbols according to their probabilities and regards them as leaf nodes in a tree. By iteratively merging the two nodes with the smallest probabilities, it creates a binary probability tree with edge labels 0 for left children and 1 for right children. Following a path from the root to a leaf provides the codeword of the corresponding symbol. By construction, infrequent symbols appear at deeper tree levels and are thus assigned longer codes than more frequent symbols. *Huffman-coding* can be shown to be optimal in regards to the entropy and is still present in many image compression codecs. However, it requires at least one bit per symbol.

The limitations of Huffman coding can be circumvented by *arithmetic coding* [234, 235, 309]. Here, the whole data is compressed at once and represented by a single number instead. This number is chosen by iteratively partitioning an interval. Each symbol of the code word corresponds to one subinterval with a length that is proportional to its probability. Thereby, each additional symbol restricts the interval further and an arbitrary number from the final interval can be selected to represent the code word. For the purpose of compression codecs, it is important to mention that there is an implementation of arithmetic coding that is believed to be patent free [189]. For example, JPEG also supports arithmetic coding in theory, but it was never implemented in commercial applications due to patents held by IBM [179].

Note that in general, not only the optimality of entropy coders plays a role for the final compression results. Also, the overhead for storing necessary information for decoding is important. For example, one needs to consider the cost for storing a Huffman tree, not only the storage needed for the actual code word.

Moreover, the entropy encoders so far have assumed that symbols can occur with the same probability at any point in a file. This is not a reasonable assumption: In an image for example, there is a high chance that a pixel has similar colours like its neighbours. So-called *higher order* entropy coding schemes incorporate such dependencies by assigning different probabilities to a symbol according to the *context* in which it appears.

There are also adaptive higher order versions of Huffman and arithmetic coding, but combining them with prediction by partial matching (PPM) [62, 195] is even more efficient. This approach just provides the probabilities for traditional entropy encoding. The encoder determines the probability of the next symbol by counting the number of occurrences of already encoded symbols in each possible context. If a symbol has not been seen by the encoder, yet, it falls back to smaller contexts. This allows to create a highly efficient encoding scheme that produces no overhead, since no additional information besides the code word needs to be stored.

The currently most successful general purpose compression method is the context-mixing algorithm PAQ [178, 180] that drives the ideas of PPM to the extreme. It consists of a large family of codecs that rely on the same principle: Many different context models for different types of data are weighted according to neural networks. These networks allow to learn and adapt the weighting throughout a file that contains many different types of data. Multiple versions of PAQ are leading coding competitions such as the Hutter prize [139] at the time of writing. Although PAQ needs a lot of memory and computation time, also a family of lightweight versions of the codec named LPAQ [180] exists. PAQ is particularly popular in PDE-based compression [183, 252].

As an alternative or additional measure to higher order methods that directly model the local dependencies of symbols by conditional probabilities of contexts, one can also decorrelate the data with *precoding*. A simple possibility is run-length encoding (RLE) [45], where the number of equal symbols that occur in an uninterrupted sequence, a so-called run, are encoded. There are many variants of this basic idea, some of which only support limited run lengths or apply entropy coders that can support infinite source alphabets (i.e. $\mathbb{N}$) like Golomb-Coding [106].

In addition to the classical entropy encoders and their extensions discussed before, there is also another important family of coders that is relevant for image compression. They take the middle ground between a one-to-one mapping of the source and code alphabet and the expression of the whole code word by a single symbol in arithmetic coding. To this end, they encode phrases that consist of multiple source symbols. Encoders such as LZ77 [328] and Lempel-Ziv-Welch coding (LZW) [303] create a list of already seen combinations of symbols and express the source word as a sequence of list indices. We refer to Sayood [245] for a more detailed overview of these methods.

In many practical applications such as gzip, the output of a dictionary encoder is additionally compressed by classical entropy encoders such as Huffman coding. In bzip2 [257], even more preprocessing steps are used: An invertible block sorting algorithm [42] creates repeating symbol sequences that increase the efficiency of RLE and Huffman coding.

☞ **Entropy Coding in PDE-based Compression**

Chapter 5.2.1 provides a more detailed discussion and comparison of specific entropy coders that are relevant for PDE-based compression. Many of the encoders are also supported by R-EED (see Chapter 3.3).

Many lossless image compression algorithms directly use the entropy coders from the previous paragraphs as steps in their compression pipeline. Since there are dozens of different file formats for images [200], we only discuss a few milestones. Early formats from the DOS era such as Picture Exchange (PCX) [12] just compressed the sequence of grey or colour values with RLE. With the advent of Windows, BMP (Bitmap Image File) [55] also added Huffman coding as an additional option for entropy coding. In 1987, the graphics interchange format (GIF) [63] replaced PCX and BMP since it offered the more efficient LZW encoding. However, patent claims on LZW lead to the development of Portable Network Graphics (PNG) [35, 68], which was explicitly designed to be a patent-free alternative to GIF. It predicts the value of a pixel from three already encoded neighbouring pixels and stores the error with DEFLATE [74], a combination of dictionary and Huffman coding. Nowadays, since the patents on LZW have expired, both GIF and PNG are popular.

Finally, the JPEG family also offers a dedicated lossless codec, JPEG-LS [301], which is technically completely unrelated to the transform-based coders discussed in Section 3.2.2. Instead, it combines prediction with context-based encoding of the prediction errors. Although JPEG-LS offers the best performance of the entropy coders discussed in this chapter as demonstrated by Fig. 3.8, it was never widely adopted. Nevertheless, it has some applications in areas where a lot of data has to be stored losslessly, such as medical imaging [202].

In addition to the lossless encoding of images with arbitrary content, there are also compression techniques that are tailored specifically to binary, i.e. monochrome images. These codecs were originally designed for fax machines and copiers and contain many features that are specific for this use. However, codecs like JBIG [146], JBIG2 [131] and DjVU [31] are also important in PDE-based compression for the storage of so-called inpainting masks (see Chapter 5).

3.2.2 Transformation-based Compression

During the 1980s, many researchers realised that the increasing popularity of personal computers and other consumer electronics would lead to a need for fast and efficient transmission and storage of digital images [215, Chapter 19]. Therefore, three international organisations for standardisation, namely the International Organisation for Standardisation (ISO), the International Telegraph and Telephone Consultative Committee (CCITT), and the International Electrotechnical Commission (IEC) cooperated in a long process of research, evaluation and consolidation that lead to the JPEG standard [215].

JPEG is a lossy codec that removes information from the image to achieve a higher compression ratio than lossless methods. It aims to remove small-scale details that are not relevant for human perception, so-called *irrelevancies*, first. Codecs from the class of *transform coders* achieve this goal by applying a transformation to the image that allows to easily distinguish between large-scale structure and small-scale detail. In the case of JPEG, a discrete cosine transform (DCT) fills this role.

For a given image $\boldsymbol{f} \in \mathbb{R}^{n \times n}$ with resolution $n \times n$, the DCT is simply a basis transform that does not represent $\boldsymbol{f}$ as a collection of pixels, but as a weighted sum of cosine waves. It is given by

$$c_{u,v} = \alpha(u)\alpha(v) \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} f_{i,j} \cos\left(\frac{(2i+1)u\pi}{2n}\right) \cos\left(\frac{(2j+1)v\pi}{2n}\right), \tag{3.69}$$

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{n}}, & \text{if } k = 0 \\ \sqrt{\frac{2}{n}}, & \text{otherwise} \end{cases}. \tag{3.70}$$

The definition above clearly illustrates that each DCT coefficient $c_{u,v}$ contains a contribution from all pixels $f_{i,j}$, thus explaining the global influence of each coefficient. Since the coefficients that correspond to high frequencies of the cosine basis functions contain small-scale structures, they are well-localised in the *frequency domain*. Compression codecs can thereby remove irrelevancies by quantising high-frequent transform coefficients more coarsely than low-frequent coefficients.

In the concrete case of JPEG, the encoder first partitions the image into 8×8 blocks before applying the DCT. This simultaneously decreases the runtime of the algorithm and allows to respect the fact that content can vary tremendously in different areas of an image. The choice of non-zero basis coefficients can be locally adapted to the image (e.g. using primarily horizontal waves in one image part and mostly vertical ones in another.) Figure 3.5 shows the approximation of one image block by a small amount of 8×8 DCT basis-images. The same figure also illustrates a severe drawback of the block decomposition: At the borders between blocks distracting discontinuities can arise, so-called block artefacts.

Key Idea: transform-based compression

In transform-based compression, images are represented in a suitable domain that decomposes the image into coarse and fine scale structures. This results in a sparse representation that can be gradually reduced by leaving away visually unimportant data.

JPEG2000 is the direct successor of JPEG and relies on the same core idea. Still, a transformation is applied, but this time with a discrete wavelet transform (DWT) instead of a DCT. As the name indicates, wavelets also exhibit wavelike properties like the cosine basis in the DCT, but also offer some distinct advantages. Unlike the cosine waves, they are designed with a limited support and do not have a global influence. In fact, the size of the support is inversely proportional to the frequency. Therefore, small-scale detail is better localised than large-scale structures. While this removes the need for block decompositions, JPEG2000 also allows so-called *tiling*. Here the image is partitioned not primarily for localisation, but for increased performance.

The core functionality however remains the same: JPEG2000 quantises the wavelet coefficients depending on their corresponding frequency and stores them with an entropy encoder, usually Huffman or arithmetic coding. While block artefacts can still occur since the wavelet coefficients are localised to rectangular supports, they are less pronounced than in JPEG since there is no fixed partitioning into 8×8 blocks.

Although the performance of JPEG2000 is in most cases superior to JPEG (see Figure 3.8), it is not widely accepted. There are several reasons for this: JPEG2000 is slower than JPEG, parts

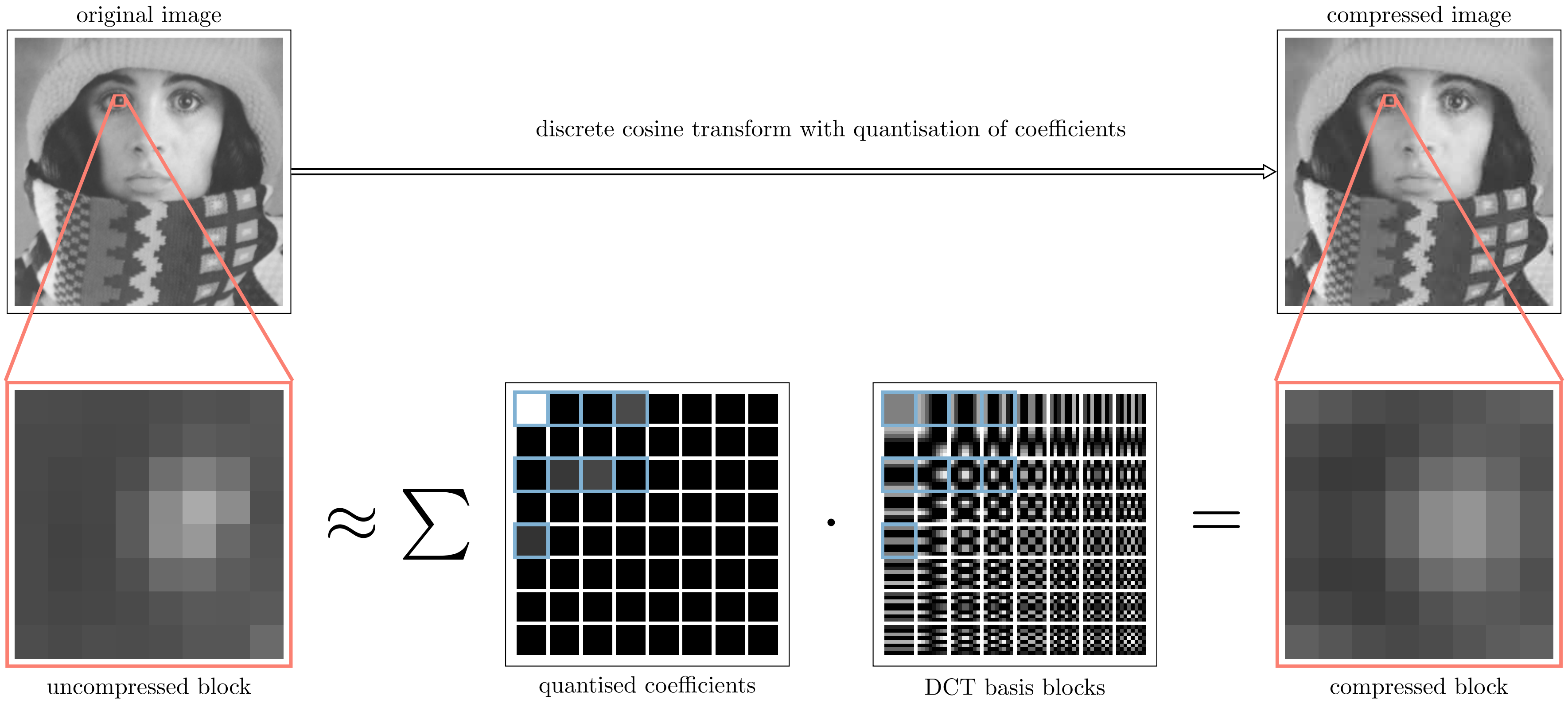


Figure 3.5.: **Transform-based Compression.** The diagram above shows how JPEG compresses an image with a combination of a discrete cosine transform and quantisation. One 8×8 block is shown enlarged below the original image. This block is approximated by the weighted sum of 9 DCT basis blocks with non-zero coefficients (marked in blue).

of JPEG2000 need to be licensed, and the quality depends significantly on the implementation of the encoder. The fact that JPEG2000 has not replaced its predecessor contains a valuable lesson. Already in 1992, Pennebaker and Mitchell [215] noted that the breakthrough in JPEG was not that it was necessarily the best image compression method, but the standardisation and independence from licensing allowed interoperability of many devices - and still does so today.

Colour Images and JPEG

Both JPEG and JPEG2000 exploit the fact that the human visual system is more sensitive to structural information than to accurate representation of colour. Thereby, both codecs decorrelate the image channels by using a YCbCr colour space. While the codecs compress all three channels by quantisation of transform coefficients, the brightness information is allowed to occupy larger amounts of disk space. This concept is discussed in more detail in Chapter 8.2, where a colour mode for PDE-based compression is introduced.

Current video compression standards like MPEG4 [143] or HEVC [271] still rely on the same basic ideas like JPEG. They are, however, a lot more sophisticated and have a lot of features that are specific to video sequences, such as removing redundancies in the temporal dimension. Interestingly, the still-image part of HEVC is competitive to JPEG2000 and Fabrice Bellard [88] proposed to use a dedicated version of this codec in BPG (Better Portable Graphics) as yet another potential successor for JPEG. Since this thesis focusses primarily on image compression, we refer the reader to Abomhara et al. [3] and Sullivan and Wiegand [270] for more information on video compression.

In general, there is a very large amount of papers that are dedicated to advancing transform-based compression. Some of this work also uses diffusion-PDEs or related variational methods in pre- or post-processing steps to improve the quality of transform coders. Note that in this thesis, we consider such techniques as *PDE-supported compression*, while the *PDE-based compression* methods in Section 3.2.3 rely on diffusion-based interpolation as the core component.

Ford [94], Yang and Hu [314], and Yao et al. [315] apply 2-D diffusion in postprocessing to remove block artefacts of transform-based coders. Alter et al. [8], as well as Bredies and Holler [38] apply a custom variational approach for removing block artefacts in JPEG that is inspired by total variation regularisation. Kopilovic and Szirányi [155] evaluate preprocessing with anisotropic diffusion [275] in the context of JPEG: Block and ringing artefacts can be reduced while the actual quantitative error only decreases moderately. Durand and Nikolova [82] propose a denoising method on Daubechies wavelet decompositions of images. This can be applied as a postprocessing step to JPEG2000 to remove Gibbs artefacts that result from setting critical coefficients to zero. In MPEG4, Bourdon et al. [33] also apply nonlinear 3-D diffusion for the same purpose. There are also approaches that rely on prefiltering instead. To this end Szirányi et al. [275] use anisotropic diffusion as a preprocessing step for image compression while Tsuji et al. [283, 284] propose preprocessing steps for video compression.

3.2.3 PDE-based Compression

Lossy PDE-based compression codecs follow the same goal as transform-based algorithms: to provide a good approximation to the original image while removing irrelevancies. Even the general idea of sparse image representations is the same. The key difference is, however, that PDE-based methods directly enforce sparsity in the spatial domain. Only small amounts of grey or colour values are known and the image is reconstructed with one of the inpainting methods discussed in Section 3.1.

The codecs presented in this section differ in how they approach the four pillars of PDE-based compression: *inpainting*, *selection of known data*, *representation of spatial locations*, and *efficient storage*. These areas are interdependent and a method can only be successful, if it combines suitable concepts for all of them. In Tab. 3.1, we give a detailed overview over the combinations of different strategies used in various codecs. The largest conceptual difference here can be found in the selection of known data. On the one hand, there are *optimisation-driven* methods that carefully optimise the locations of scattered data points to minimise the difference to the original. On the other hand, *feature-driven* codecs store semantic structural components of the image such as edges and try to get the maximum quality out of this given data.

Key Idea: PDE-based compression

Only a small amount of carefully selected image features is stored. Parital differential equations model an image evolution that propagates known information to missing image areas, thereby reconstructing the image.

Optimisation-Driven Codecs

Optimisation-driven algorithms have initiated the recent research activity in PDE-based compression, starting with the pioneering work of Galić et al. [100]. Their edge-enhancing diffusion codec (EEDC) combines inpainting with EED [295] and a triangular partitioning of the image that specifies the location of known data and allows to store them with a binary tree. The triangular subdivision is inspired by B-tree triangular coding (BTTC) [76], an earlier compression method that used liner interpolation instead of EED. Not only does the binary tree representation help to store data efficiently, it also constitutes the optimisation strategy. The image is successively subdivided into triangles where the local inpainting error is too large, thus increasing the amount of known data in regions where the reconstruction is not adequate.

	Year	Inpainting		Data Selection				Spatial Representation					Efficient Storage					
		Homogeneous	EED	Subdivision	Edge Detection	Probabilistic Sparsification	Source Points	Tree-Structure	Binary Image	Regular Grid	Free Points	Chain Codes	Huffman Coding	Arithmetic Coding	PAQ	LPAQ	JBIG	others
Galić et al. [99, 100]	2005	○	●	●				●					●					
Schmaltz et al. [250, 252]	2009	○	●	●				●					○	○	●			○
Mainberger et al. [181, 183]	2009	●			●				●				○	○	○	●	●	○
Wu et al. [310]	2009	●			●				●									●
Bastani et al. [20]	2010	●			●		●		●								●	●
Zhao and Du [322]	2011	●			●							●						●
Gautier et al. [103]	2012	●			●							●		●				●
Li et al. [166]	2012	●			●							●		●	●			
Hoffmann et al. [127]	2013	●			●	●			●	●	●	●			●		●	

Table 3.1.: **Comparison of PDE-based Coders.** In this table, we show the similarities and differences of a selection of PDE-based coders in regards to the four pillars of PDE-based compression: inpainting, selection of known data, representation of spatial locations, and efficient storage. Note that we only consider 2-D image compression codecs and do not include other kinds of image data (3-D, video, surfaces) or hybrid methods. The codecs are ordered chronologically after their first publication date from top to bottom. Filled circles in the table represent default features while empty circles stand for supported optional features.

Qualitywise, after some improvements of the original method by Galić et al. [99], EEDC can surpass JPEG, but is not able to consistently reach the quality of JPEG2000.

Schmaltz et al. [250] advanced the original concept of EEDC by introducing a number of novel concepts. Their R-EED codec uses rectangular subdivisions instead of triangular ones, more sophisticated parameter optimisation strategies, and postprocessing steps. In images that are not dominated by texture, this codec can surpass JPEG2000 for most compression ratios. Schmaltz et al. [252] provide additional improvements to the algorithm as well as some early examples for colour images and explanations for the superior performance of EED. At the time of writing, R-EED is still the most successful PDE-based codec for general image content. Section 3.3 explains this algorithm in more detail.

It should be noted that both EEDC and R-EED use heuristic strategies to select the known data. However, the success of PDE-based compression has sparked interest in choosing truly optimal data for PDE-based inpainting. A probabilistic approach by Mainberger et al. [184] successively reduces the amount of known pixels from the full image to a given percentage. After the inpainting mask has reached the desired number of sparsity, a non-local pixel exchange randomly swaps the position of candidate pixels and checks if the error decreases. While this stochastic approach is very costly, it cannot be caught in local minima.

For linear diffusion methods, several other optimisation strategies have been proposed. Belhachmi et al. [25] provide an analytical theory for this spatial optimisation problem in the case of homogeneous diffusion. Their results indicate that known data should be placed at locations of high Laplacian magnitude. This also motivates the success of some feature-driven codecs from the next section that rely on image edges as locations for known data. Hoeltgen et al. [124] use an optimal control approach that minimises the reconstruction error while imposing a sparsity assumption on the known data and an inpainting constraint for the reconstruction of the missing image parts. A similar model was also considered by Ochs et al. [209] as an application of their iPiano algorithm, an approach for solving partially non-convex optimisation problems. Finally, Chen et al. [59] build on the work of Hoeltgen et al. [124] and extend the model to contain not only linear homogeneous diffusion, but also a linear higher order reconstruction model, namely biharmonic interpolation. In addition, they consider efficient solvers for this problem.

☞ **Spatial Optimisation in PDE-based Compression**

It is important to note here, that all of the publications from the previous paragraph deal purely with the spatial optimisation problem. They do not propose actual compression codecs. Only the ideas of Belhachmi et al. [25] have been used in some specialised codecs that are discussed in the following sections [181–183]. We address this gap in Chapter 5.

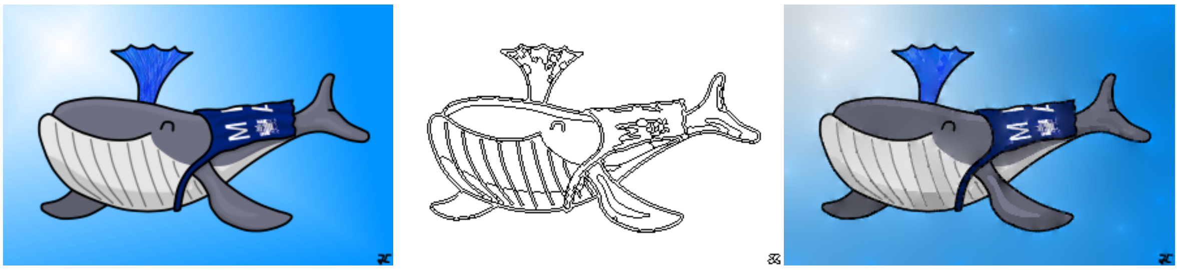
Feature-Driven Codecs

There is a long history of representing image content purely by edges. Both Werner [307] and Elder [86] have argued that edges are rich features. They are not only important for human perception and offer an intuitive representation of structure, they can also provide a lot of higher-order image information such as gradient directions or contrast changes, if pixel values in their vicinity are also stored. Since a large amount of research is dedicated to edge-based interpolation [57, 69, 79, 110, 137, 174, 186, 236, 241, 319], it is not surprising that many PDE-based compression codecs also rely on this feature. One obvious advantage over the optimisation-driven encoders is runtime: Edges can be cheaply detected, e.g. with a Canny edge detector [43] or more sophisticated methods.

For instance, Acar and Gökmen [4] use the same variational membrane model to detect edges and reconstruct the image from a sparse edge set. Similarly, a more sophisticated centipede model is the core ingredient for the compression technique by Kurt et al. [163]. It performs edge detection and segments the image. These segments are then ranked according to a confidence measure that incorporates boundary lengths and statistical moments. The codec only stores the most significant segments and reconstructs them with a variational method. Many codecs follow a standard pipeline: pre-processing to remove noise, followed by edge detection and some sort of entropy coding for compression. For decompression, linear homogeneous diffusion is employed by all of the following methods. Carlsson [47] proposed an early sketch-based approach with linear homogeneous diffusion which was later modified and extended by Desai et al. [73]. The codec for general images by Wu et al. [310] performs edge detection and edge extension. It then stores the grey values at thickened edges with JPEG2000 and reconstructs with homogeneous diffusion. In contrast, Bastani et al. [20] store edges, but restrict grey values to so-called source points. Before edge detection, their algorithm uses Perona-Malik filtering for denoising. Similarly, Zhao and Du [322] employ a modified Perona-Malik filter for presmoothing and extract edges as zero-crossings of the Laplacian afterwards. Edges are stored with a so-called chain code that interprets an edge as a path that is traversed from beginning to end. Thereby, the edge is uniquely determined by the sequence of directions taken from one edge pixel to the next on the path.

Apart from these linear, feature-based approaches, there are some attempts to use nonlinear PDEs for image compression. Chan and Zhou [53] propose a variational approach with total variation regularisation to minimise oscillations in wavelet decompositions. Furthermore, Gomathi and Kumar [107] use a p-Laplacian for interpolation in the wavelet domain combined with neural networks. This work can be considered a hybrid approach somewhere between PDE-based and transform-based compression.

In general, feature-based general purpose algorithms have a hard time competing with JPEG and JPEG2000 qualitywise. The optimisiation-driven approaches from the previous section



cartoon whale edges reconstruction

Figure 3.6.: **Cartoon Reconstruction with PDE-based Inpainting.** From its edges, the cartoon whale on the right can be reconstructed with simple homogeneous diffusion. However, the reconstruction works only well for truly homogeneous areas. Details, e.g. the painting strokes in the sprouting water or the logo on the cape are lost. Images courtesy of J. Contelly and M. Puhl [229].

have a clear advantage in this regard, since they actively incorporate the error measure into the compression process. However, feature-based approaches can still produce a very high quality for special classes of image content. In these cases, they are highly competitive to transform-based coders, because they excel not only in reconstruction quality, but also in speed.

Mainberger et al. [181, 183] have shown that simple homogeneous diffusion can beat JPEG 2000 on cartoon-like images. They extract image edges, store their locations with JBIG and take grey or colour values left and right of these edges as known data. Since these pixel values are highly redundant, they can be sampled coarsely along the contours and are efficiently encoded by an entropy encoder like LPAQ. The availability of exact edge data circumvents the main drawback of homogeneous diffusion, its inability to reconstruct sharp contrast changes (see Figure 3.6).

This also makes homogeneous diffusion well-suited for depth map compression, because this data is naturally composed of piecewise smooth image regions. Gautier et al. [103] and Li et al. [166] use edge features and homogeneous diffusion similarly as the approach of Mainberger et al. [183] with specific adaptations to depth maps. Hoffmann et al. [127] go beyond pure storage of edge data and partition the depth map into non-overlapping regions. By enforcing Neumann boundary conditions on the segment borders, the reconstruction can contain sharp edges without the need to store all pixel values left and right of the edges as in cartoon compression [183]. Instead, known data is given on a regular, hexagonal grid which requires to store no positional data except for the grid size. Since carefully chosen known pixels can significantly increase the reconstruction quality of linear diffusion, Hoffmann et al. also choose and store additional free points with a stochastic approach [184]. Thereby, this is a hybrid approach that is both optimisation- and feature-driven.

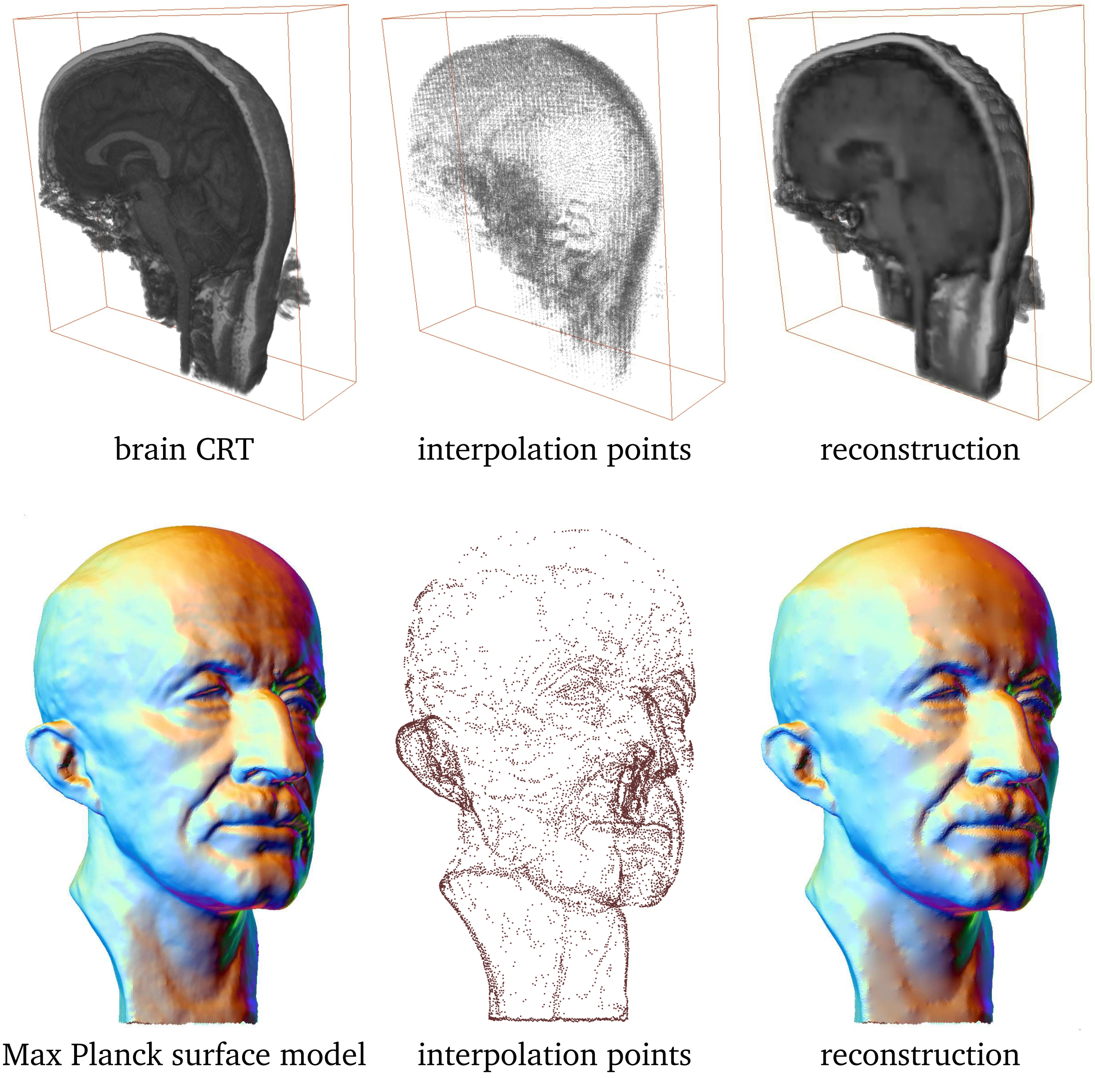
In addition to the contour-based methods, there are also point features that can be used for compression. Johansen et al. [145] and Kanters et al. [149] reconstruct images from so-called top points, for which first and second order derivatives vanish. These points are classically used as descriptors of the deep structure in scale-space theory. Kanters et al. [149] use a minimum variance scheme to reconstruct images based on a natural image prior, the so-called Brownian reconstruction. Other approaches use local jets [167], SIFT features [302] or binary descriptors [70]. However, as Hoeltgen et al. [125] remark in their review on PDE-based compression, sparse point-based features usually do not provide adequate known data for compression. Early experiments with PDE-based reconstructions from corners [326] yielded promising results that however stayed behind the quality of JPEG.

Feature-based coding can also be used for geometric shapes. Huth [138] and Schmaltz et al. [252] specify quadrupoles along the contour of shapes. A quadrupole is a 2×2 block of pixels. Its grey values encode the information that EED needs to reconstruct the shape boundary. The codec chooses the number of quadrupoles proportional to the average curvature of the contour. Extensions of these ideas to nonapoles were explored by Steil [265].

Extensions and Applications

One general drawback of PDE-based methods is that they are not well-suited for textured content. Small scale texture is not well represented by the underlying smoothness assumption of PDE-based inpainting. Therefore, combining PDE-based with other interpolation techniques seems viable. Liu et al. [169] integrate inpainting into existing codecs and supplement it with texture synthesis. Zimmer [327] and Schwinn [255] have also experimented with texture inpainting techniques that could supplement PDE-based compression. In this thesis, we present a hybrid approach that reconstructs structure with PDE-based inpainting and texture with patch-based interpolation (see Chapter 6).

Video coding with PDEs is still in a very early development stage. The only full-fledged codec so far is a model-based approach for video compression by Schmaltz and Weickert [249]. It is designed for predominantly static scenes in which well designed objects move, e.g. video conferencing. Here, the objects are first tracked and then replaced by a 3-D model, while the static background is encoded with R-EED-like image compression. Furthermore, Köstler et al. [156] have shown that video decompression with a resolution of 320×240 is possible on the parallel architecture of a Playstation3. Baum [22] has also presented a real-time video decompression technique with a demonstration codec that uses sparsification to determine image points and a DEFLATE container for compression. However, the codec sacrifices quality in order to achieve realtime performance. In addition, there are also hybrid approaches that do not purely use PDEs for regularisation, but still rely primarily on transform coding: Gao [101] compute the so-called optic flow, a displacement vector field, between subsequent frames



brain CRT

interpolation points

reconstruction

Max Planck surface model

interpolation points

reconstruction

Figure 3.7.: **3-D Reconstructions with PDE-based Inpainting.** The first row shows 3-D voxel reconstructions with EED in medical imaging by Peter [218], while the second row displays an example of surface reconstruction by Bae and Weickert [14].

for motion compensation in transform-based coding. This method stores the smooth flow fields by sparse masks and diffusion-based inpainting. Similarly, Doshkov et al. [77] use linear homogeneous inpainting for prediction in transform-based video compression.

There is also some work on specialised PDE-codecs for different kinds of data. For 3-D voxel data which occurs e.g. in medical data, Lund [177] has investigated compression with homogeneous diffusion. Peter [218] introduced a cuboidal subdivision scheme with edge-enhancing diffusion that extends R-EED by using redundancies in all three dimensions. Concerning a different kind of 3-D data, Bae and Weickert [14] compress geometric mesh surfaces by storing only a few vertices and reconstruct the full surface with a geometric PDE. Figure 3.7 contains two examples for 3-D reconstruction with diffusion. Finally, elevation maps that contain level lines can also be compressed with PDE-based inpainting [89, 311]. For instance, Solé et al. [262] use the Laplacian, the biharmonic operator, and AMLE [11].

In regards to transmission over limited channels, Schmaltz et al. [251] have introduced progressive modes for R-EED. By splitting up the tree structure and ordering the stored pixel values in a suitable way, a hierarchical image structure is created. The progressive codec can already reconstruct a preview version of the image after a small amount of data has been transmitted. This representation can then be refined multiple times as soon as more data is available.

In addition, there are also applications in cryptography that are broadly related to PDE-based compression. Steganography deals with the task to conceal a secret image inside of a cover image. Mainberger et al. [182] use a fast PDE-based codec that relies on dithering of a Laplacian magnitude map to find known data (a heuristic inspired by [25]) and reconstructs the hidden image with EED. The known data are stored as small perturbations of the cover image's pixel values. This approach has also been extended by Niu [205]. Similarly, Ou et al. [212] use PDEs for prediction in lossless hiding of information.

So far, there have been almost no considerations of specialised entropy coding for PDE-based compression. Most codecs use already existing algorithms. Scheer [247] has shown that even tailor-made prediction methods have a hard time competing with learning approaches like PAQ.

Finally, there are a few publications that are broadly related to the aforementioned PDE-based methods, but consider different kinds of data or employ other PDEs. Both Brinkmann et al. [39] and Schneider [254] store gradient data instead of only saving grey values. Schneider [253] has considered diffusion on level-lines.

3.2.4 Other Compression Approaches

In addition to the transform coders and PDE-based methods for image compression, there is still a wide variety of other approaches. Kunt et al. [162] and Reid et al. [233] both provide surveys on the field of *second-generation coding*. This research area covers all compression techniques that do not purely rely on redundancy removal by entropy coding. In the following, we briefly mention a small selection of methods that are broadly related to the PDE-based approaches discussed before.

Besides the BTTC approach by Distasi et al. [76] that inspired EEDC and R-EED, there are a lot of other techniques that use tree representations [13, 152, 266, 267, 269, 286]. The very general concept of subdividing the image domain and using this subdivision to represent locations as a tree can be combined with many different compression ideas. For example Distasi et al. [76] apply simple linear interpolation, while Aurich and Daub [13] use a custom edge-preserving technique. Other techniques combine transform-based ideas with trees [152]. There are also depth map compression algorithms that use either triangular [244] or quad trees [197].

These methods are also related to triangulation ideas. Demaret and Iske [72] as well as Demaret et al. [71] use thinning approaches to reduce the image to a small amount of significant pixels. Decompression is performed by linear interpolation over the Delauney-triangulation of the significant pixel set. Similarly, Bougleux et al. [32] use an anisotropic triangulation technique to obtain a mesh that acts as an inpainting mask. From the vertices, each triangle in the mesh is reconstructed with spline interpolation.

Furthermore, there are also edge-based inpainting approaches for depth maps that do not use diffusion, but other interpolation techniques. This holds also true for edge-based compression of depth maps by Chen et al. [56] and Jäger [144]. Moreover, there is also the depth map compression approach by Förster et al. [95] that incorporates information of the corresponding colour image into the compression codec. This idea can be seen as broadly related to the luma guided diffusion we propose in Chapter 8.2.

In video compression, there are codecs that are related to the model-based approach by Schmaltz and Weickert [249]: For instance, Toelg and Poggio [279] have proposed a video conferencing codec that uses example images of faces for more efficient compression.

Finally, fractal image compression [111] is a completely different approach to the previously discussed ones. The underlying assumption here is that images are highly self-similar. The image is represented by a collection of regions that do not cover the whole image. Missing regions are approximated as simple transformations such as rotations, mirroring and rescaling of the known regions. Here, the challenge is to find these self-similarities in a very large search space. Fractal image compression is related to exemplar-based inpainting that is discussed in Chapter 6. Such patch-based methods fill in missing values according to known data that has a similar pixel neighbourhood.

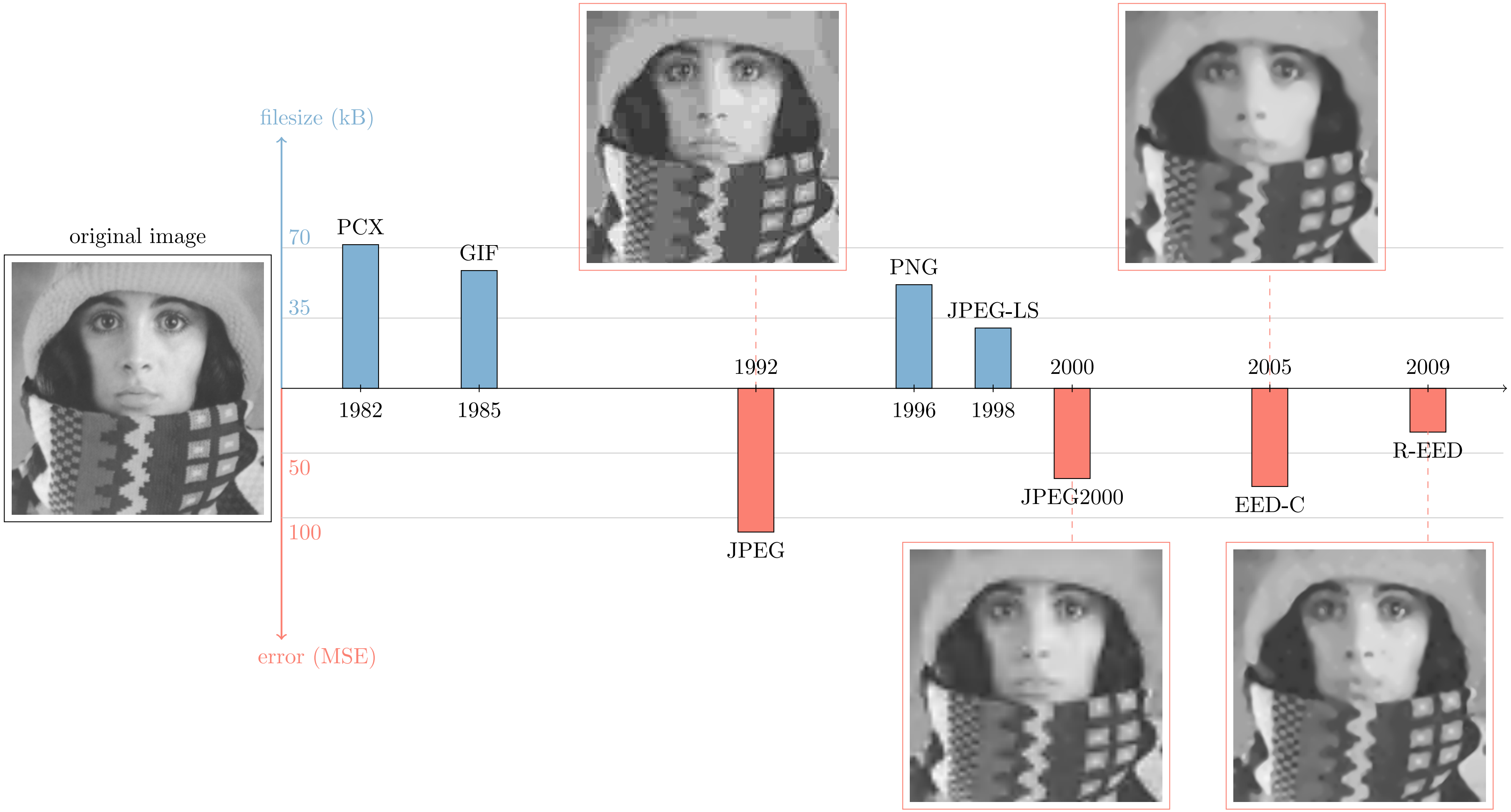


Figure 3.8.: **Evolution of image compression.** The timeline shows a selection of important milestones in compression. The upper part of the chart displays the file size of the test image *true* for lossless compression codecs in blue. MSE values for transform- and PDE-based methods at a compression ratio of 56:1 are denoted by the red bars below the timeline.

3.3 PDE-based Compression with R-EED

> *Today is yesterday's pupil.*
>
> — **Benjamin Franklin**
> (politician, author, and scientist)

Since the R-EED algorithm by Schmaltz et al. [250] is currently the most successful general purpose encoder based on diffusion PDEs, it is a good starting point for advances in the field. Therefore, we describe the codec in more detail in the following paragraphs.

R-EED belongs to the optimisation-driven compression codecs from Section 3.2.3: It chooses the locations of known data in such a way that the corresponding reconstruction approximates the original image. However, it does not only take the reconstruction quality into account, but also the cost of encoding the locations of each point. Its name stems from the two main ingredients that enable it to beat JPEG2000: the rectangular subdivision strategy for mask selection and the edge-enhancing anisotropic diffusion (EED) inpainting that it uses to reconstruct the image from the known data.

Key Idea: R-EED

R-EED selects and stores known data efficiently by subdividing the image into smaller subpictures. This subdivision can be represented by a binary tree and is guided by the quality of local reconstructions with EED.

Adaptive Rectangular Subdivision

As discussed in our review of PDE-based compression, it is not only important to choose a good operator such as edge-enhancing anisotropic diffusion for inpainting. In addition, adequate known data has to be selected in order to obtain good image quality [25, 59, 124, 184, 209]. However, choosing such optimal data is non-trivial. There is a limited number of possibilities to choose pixels from an image with resolution $n_x \times n_y$. Already for a 512×512 image, $9.5 \cdot 10^{37006}$ different masks with a density of 10% exist. In addition, scattered data is expensive to store. The positional data is essentially equivalent to a binary image with the full resolution of the original.

Instead of using free points for the inpainting mask, Schmaltz et al. [252] restrict the choice of mask locations to an adaptive rectangular grid. This restriction solves both aforementioned problems: On one hand, the locations become easier to store since they are well-structured. On the other hand, the size of the search space for known data is significantly reduced. Overall, such a constrained mask that might be suboptimal from the view of pure inpainting can be

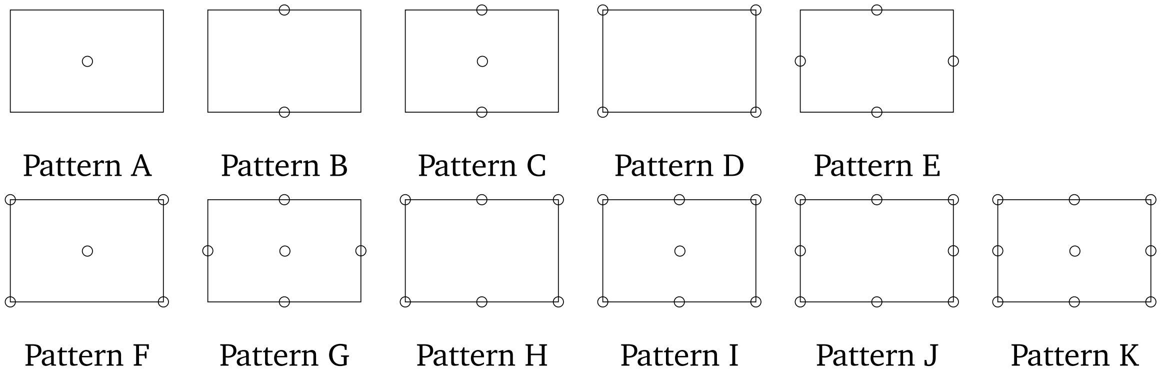


Figure 3.9.: **Point patterns in R-EED.** For every partial image in rectangular subdivision, R-EED adds known data according to the patterns above. Pattern F has been proven to perform particularly well. Image courtesy of Schmaltz et al. [252].

better than an optimal mask, if one considers the trade-off between file size and reconstruction quality.

An adaptive grid can be achieved by specifying a fixed point pattern for rectangular images. The position of known points is defined relative to the image boundary. Schmaltz et al. [250] have determined empirically that the four corners and the midpoint lead to a good distribution of data for subdivision schemes by testing a lot of different point patterns (see Figure 3.9). In order to obtain an adaptive grid, we add the known data of the selected point pattern to the image to compute a preliminary inpainting result.

If the error is too high, the algorithm splits the image in half in its longest dimension, yielding two rectangular subimages. For each of these subimages, we then repeat the whole procedure: adding the data, performing a localised inpainting, and investigating the error. Doing this successively refines the grid in regions where the local error is high. Each of the splits can be interpreted as one node in a binary subdivision tree (see Figure 3.10).

Since the subimages become smaller after each split, but the MSE is normalised in regards to the number of pixels, Schmaltz et al. [250] use an adaptive error threshold $T := a\ell^d$. Here, the global threshold a specifies the maximum error on the full image. The additional factor ℓ adapts the error threshold exponentially to the tree depth d.

Note that for a reconstruction of the original image, the decoder needs to know the positional data represented by the binary tree and the pixel values that were selected by the point pattern. The tree can be represented by a bit sequence that indicates if a node is split or not. Schmaltz et al. [250] store this sequence verbatim in the file header without additional compression.

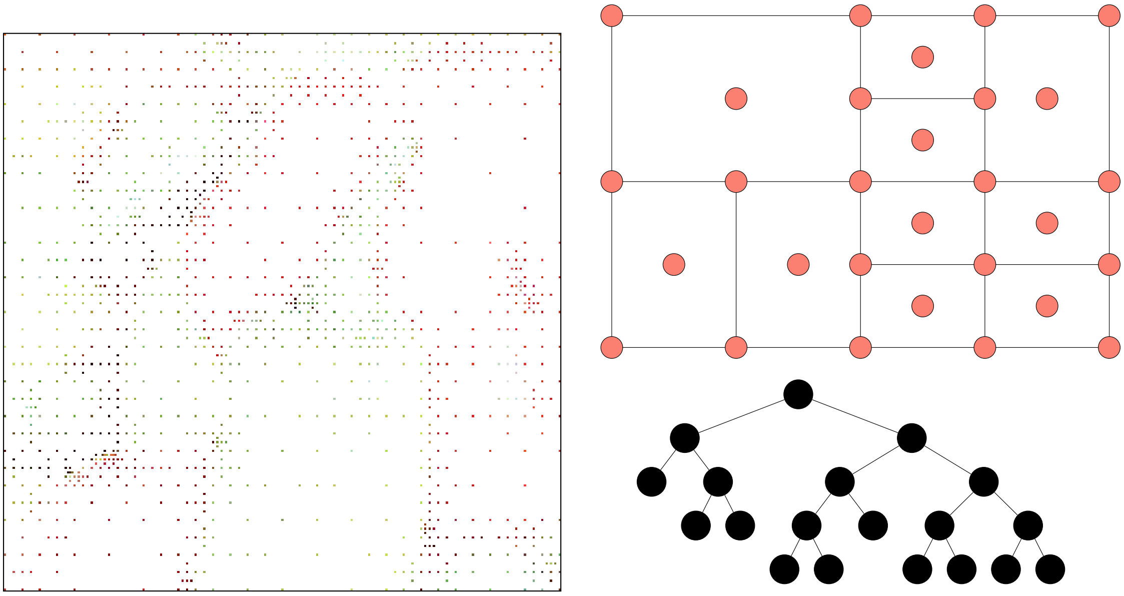


Figure 3.10.: **Subdivision in R-EED**. The known data for the image *peppers* on the left hand side lives on a regular adaptive grid. An example on a smaller image is given on the right hand side: Known data points on the grid are signified by red dots. This subdivision can be equally characterised by the binary tree below. Each leaf node in the tree corresponds to one subimage. A more detailed discussion of the subdivision that leads to this example can be found in Figure 5.2 of Chapter 5.

☞ **Tree Representations in PDE-based Compression**

In Chapter 5 we give some more details on tree representations of spatial locations. Furthermore, we investigate how binary trees can be stored more efficiently than in plain bit sequences and investigate a probabilistic strategy for tree-based rectangular subdivisions.

Entropy Coding

The grey values are first uniformly quantised by R-EED. Since PDE-based inpainting can reconstruct also grey values that lie in-between the quantised pixel data, coarse quantisations around 30 to 40 individual values are possible.

After quantisation, an entropy coder removes redundancies in the sequence of pixel values that need to be stored. Schmaltz et al. [250] have compared many of the different entropy encoders discussed in Section 3.2.1, namely Huffman coding, LZW, gzip, bzip2, arithmetic coding, and PAQ. In their experiments, PAQ and arithmetic coding yielded the best compression results.

Postprocessing

R-EED allows two optional postprocessing steps after rectangular subdivision: *grey value optimisation* that modifies the stored pixel values and *mask inversion* that determines, if an additional inpainting step during decoding can improve the overall quality.

Multiple publications [59, 125, 126, 184] have shown that selecting not only optimal positions, but also optimal grey values can increase the inpainting quality of PDE-based compression. This implicates that one introduces an error to the known data. However, since the known data only covers a small percentage of the whole image domain, the gain in reconstruction quality for the missing areas can outweigh the loss in accuracy at mask positions. As a positive side effect, grey value optimisation can also correct for noisy known data.

In R-EED, grey value optimisation is performed sequentially for each point of the inpainting mask in random order. The algorithm tests, whether the global inpainting error decreases, if it replaces the pixel value by either the next higher or next lower value from the quantised co-domain. If the error becomes worse, the original grey value is preserved. R-EED iterates this process until convergence.

For the second optional postprocessing step, the algorithm switches the roles of the inpainting mask K and the inpainting domain $\Omega \setminus K$. The new inpainting domain $\Omega \setminus K'$ consists of circles with radius r around all points of the former inpainting mask K. These areas are now considered to be missing while all other image regions from the original inpainting domain are treated as known data. The encoder checks now, if the inpainting error can be reduced with this new configuration. Such a postprocessing step can reduce errors in the known data that stem from lossy steps such as quantisation or grey value optimisation.

Compression and Decompression Pipeline

In summary, the core of the R-EED encoder is the subdivision algorithm. It interleaves splitting and inpainting steps until the error is smaller than the threshold T in all subimages or the maximal depth of the tree has been reached. Next, the tree is encoded as a binary string and the known pixel values are selected according to the point pattern p.

After the grey value domain has been reduced to q different values by quantisation, the list of pixel values can be stored losslessly with entropy coding. Grey value optimisation can be performed either before or after grey value encoding. Note however that the size of the file might change slightly after the grey values have been modified and entropy coded again.

For decoding, the algorithm first reconstructs the locations of the inpainting mask according to the binary tree and the point pattern p. Afterwards, the pixel values are recovered by reversing the entropy coding. Finally, the image can be reconstructed with EED inpainting. As a last step, R-EED performs the optional mask inversion and an additional inpainting if necessary.

Parameter Optimisation

The previous paragraphs have introduced several different free parameters for R-EED. In order to obtain a good quality for high compression ratios, all of these parameters have to be adjusted to each individual image. Since all of the following parameters are co-dependent, i.e. tuning one parameter influences the choice for all other parameters, many combinations must be considered. The reference implementation of Schmaltz et al. [250] optimises the following parameters with a multi layered binary search. This comes down to building a full binary tree for each parameter combination. To a certain degree, the computational load can be reduced by caching the error values of subimages.

Rectangular subdivision: The point pattern p is fixed, but nevertheless, two parameters from the formula $T := a\ell^d$ need to be optimised.

Inpainting: For EED inpainting, the algorithm optimises the contrast parameter λ. According to Schmaltz et al. [250], the influence of the presmoothing scale σ is small, if σ has a reasonable value and λ is chosen optimally. In practice, they use a fixed value $\sigma = 0.8$.

Quantisation: Here, only the number $q \leq 256$ of grey values is a free parameter. Schmaltz et al. [250] choose it in such a way, that a given tree obtains the compression ratio specified by the user. If the ratio cannot be reached for a given combination of the parameters a, ℓ, and λ, the corresponding tree is simply discarded.

In addition to these four parameters, a manual limitation of the minimal and maximal tree depth can improve the compression performance. The user also preselects the entropy coder (the default is arithmetic coding). During grey value optimisation, R-EED interleaves each pass on all grey values with a new optimisation of the contrast parameter λ. Finally, the global radius r of the circle for mask inversion can be determined by a simple brute force search, since only very small, integer-valued radii are feasible.

4 Justifying Tensor-Driven Diffusion from the Statistics of Natural Images

> *If you torture the data enough,*
> *nature will always confess.*
>
> — **Ronald Coase**
> (Economist and Nobel Prize Winner)

Chapter 3.1 has already introduced anisotropic diffusion models that involve a positive definite 2×2 matrix called diffusion tensor. Its eigenvalues steer the amount of data propagation in the direction of the corresponding eigenvector. We have discussed its relevance for inpainting (e.g. [281]) and compression both in relation to prior work (e.g. [99, 252]) and to this thesis. In addition, throughout more than two decades of research, such anisotropic methods have been successfully used for a large number of image processing and computer vision problems beyond compression. These tasks include denoising [64], optical flow computation [201], stereo reconstruction [325], and shape from shading [6]. Application domains cover e.g. computer aided quality control [292], medical image processing [187], and seismic image analysis [123].

To this date, modelling nonlinear diffusion filters is a heuristic, task-driven procedure, where images are processed towards a certain goal. We have illustrated this problem-centric strategy in our discussion of denoising and inpainting with prototypic diffusion filters in Chapter 3.1. Our experiments in the aforementioned chapter demonstrate the well-known fact that anisotropic models can be much more powerful in certain applications than isotropic diffusion approaches with a scalar-valued diffusivity function. Clearly, one reason for the success of anisotropic concepts are their additional degrees of freedom, which can be adapted to the task at hand. However, another potential explanation for this success is still unexplored: Could it be that smoothness assumptions of anisotropic models reflect statistical properties of natural images more accurately than isotropic ones?

For specific *isotropic* diffusion models, there exists a well-known connection to probabilistic filter models based on the statistics of natural images [323]: There is a negative logarithmic correspondence between natural image priors and regularisation terms in variational models. However, in particular for *anisotropic* diffusion, previous investigations have focused on isolated, specific models in practical contexts such as parameter learning. In particular, there is a lack of a cohesive theory that systematically analyses the correspondence between probabilistic filters and diffusion filters that can be expressed by energy minimisation.

Our Contributions. In this chapter, we have the goal to provide a justification of tensor-driven diffusion models via the statistics of natural images. We aim at systematically assessing the differences between isotropic and anisotropic approaches from a probabilistic perspective.

To this end, we use natural image priors to derive a unifying framework incorporating eight existing diffusion filters that have a corresponding variational formulation. In order to cover the full range of nonlinear models, these statistics have to reflect the local image structure and allow to involve directional information. The eigenvalue statistics of the structure tensor in databases of natural images provide not only such information, but also offer a lot of flexibility to generate a wide range of derivative-based priors. This allows us to construct probabilistic filters that represent existing isotropic and anisotropic filter classes and analyse the differences in the underlying priors. We discuss the implications of these differences on filter performance in the context of image denoising.

Moreover, we investigate if the Bayesian priors can be used for parameter learning in image inpainting or compression applications. We also consider perspectives for future additions to our probabilistic framework, such as edge-enhancing anisotropic diffusion.

Organisation of the Chapter. We start with a brief overview of related work and mathematical foundations that are relevant only for this chapter. Section 4.1 provides a short review of important tensor-driven models. With Section 4.2, we establish the probabilistic background for our new framework and discuss prior work about the relations of diffusion and Bayesian models.

In Section 4.3, we investigate the properties of the structure tensor as an image feature and use it to derive a probabilistic denoising filter in Section 4.4. We show that this model is related to a unifying framework for diffusion filtering in Section 4.5. In Section 4.6 we investigate diffusion models that are learned from a database, evaluate their performance for denoising, and interpret the results.

While the previous sections all deal with general results on the connections between statistics and filter design, Section 4.7 and Section 4.8 focus on the implications of this chapter on the practical compression applications in the rest of the thesis. Finally, we present our overall conclusions for this chapter in Section 4.9.

4.1 Tensor-Driven Diffusion Processes

> *I can't change the direction of the wind, but I can adjust my sails to always reach my destination.*
>
> **— Jimmy Dean**
> (Actor and Musician)

Let us start by reviewing a number of isotropic and anisotropic diffusion filters which can be derived from a general energy functional that we have presented in Chapter 3.1. In addition to the three prototypic models discussed there, we also deal with diffusion processes that incorporate Gaussian derivatives or modifications for colour images. Note that we consider edge-enhancing anisotropic diffusion (EED) separately in Section 4.8.

General Structure. Let $\boldsymbol{f} = (f_1, ..., f_{n_c})^\top$ represent a vector-valued image with n_c channels. Each of these channels is a function $f_k : \Omega \to \mathbb{R}$ that maps the rectangular image domain $\Omega \subset \mathbb{R}^2$ to the colour value range $\mathbb{R}$. A tensor-driven, vector-valued diffusion process computes filtered versions $\{\boldsymbol{u}(x,y,t)\,|\,(x,y) \in \Omega,\ t \geq 0\}$ of $\boldsymbol{f}(x,y)$ as solutions of the diffusion equation

$$\partial_t u_k = \operatorname{div}(\boldsymbol{D}\,\boldsymbol{\nabla} u_k) \quad \text{on} \quad \Omega \times (0,\infty), \quad k = 1, \ldots, n_c \tag{4.1}$$

with $\boldsymbol{u}(x,y,0) = \boldsymbol{f}(x,y)$ as initial condition on Ω, and reflecting boundary conditions:

$$\langle \boldsymbol{D}\,\boldsymbol{\nabla} u_k, \boldsymbol{n}\rangle = 0 \quad \text{on} \quad \partial\Omega \times (0,\infty), \quad k = 1, \ldots, n_c. \tag{4.2}$$

The diffusion time t serves as a scale parameter: Larger times yield simpler image representations. The nabla operator $\boldsymbol{\nabla}$ and the divergence operator div involve spatial derivatives only, and $\boldsymbol{n}$ denotes the outer normal vector to the image boundary $\partial\Omega$. The diffusion tensor $\boldsymbol{D}$ is a positive definite 2×2 matrix that steers the diffusion. Its eigenvalues specify the amount of diffusion in the direction of the corresponding eigenvectors.

Isotropic Models. The simplest diffusion process, *homogeneous* diffusion [141], is obtained for $\boldsymbol{D} := \boldsymbol{I}$ with a unit matrix $\boldsymbol{I}$. In this case, the diffusion does not depend on the image structure. For more sophisticated *nonlinear isotropic* diffusion models the diffusion tensor is of the form $\boldsymbol{D} := g(|\boldsymbol{\nabla} u|^2)\boldsymbol{I}$. Thus, the tensor degenerates to a scalar *diffusivity* $g(|\boldsymbol{\nabla} u|^2)$ and one obtains the isotropic diffusion equation

$$\partial_t u = \operatorname{div}(g(|\boldsymbol{\nabla} u|^2)\,\boldsymbol{\nabla} u) \quad \text{on} \quad \Omega \times (0,\infty). \tag{4.3}$$

If one wants to permit strong smoothing within homogeneous regions and inhibit smoothing across edges, one chooses the diffusivity as a decreasing positive function of its argument.

Many diffusivity functions have been proposed, e.g. the Perona/Malik diffusivity g_{PM} [217] or the Charbonnier diffusivity g_{C} [54]:

$$g_{\mathrm{PM}}(s^2) := \left(1 + \frac{s^2}{\lambda^2}\right)^{-1}, \qquad g_{\mathrm{C}}(s^2) := \left(1 + \frac{s^2}{\lambda^2}\right)^{-1/2}. \tag{4.4}$$

Note that locations where $|\boldsymbol{\nabla} u| \gg \lambda$ are regarded as edges where the diffusivity is close to 0, while we have full diffusion in regions with $|\boldsymbol{\nabla} u| \ll \lambda$. Therefore, $\lambda > 0$ acts as a contrast parameter. Isotropic models allow space-variant smoothing, but due to their scalar-valued diffusivity, the diffusion process acts in the same way in all directions.

So far, we have only considered greyscale images ($n_c = 1$) for isotropic diffusion. This covers for instance the first isotropic nonlinear model that goes back to Perona and Malik [217] and was already introduced in Chapter 3.1. Gerig et al. [104] have extended this approach to colour image processing by coupling the evolution of the individual channels through a diffusivity of the form $g(\sum_{k=1}^{n_c} |\boldsymbol{\nabla} u_k|^2)$. Moreover, Scherzer and Weickert [248] have investigated an isotropic nonlinear diffusion model where all spatial gradients $\boldsymbol{\nabla}$ are replaced by Gaussian-smoothed gradients $\boldsymbol{\nabla}_\sigma := K_\sigma * \boldsymbol{\nabla}$. Here, K_σ is a Gaussian with standard deviation σ.

Anisotropic Models. In order to model direction-dependent diffusion processes, we need an anisotropic diffusion tensor $\boldsymbol{D}$ whose eigenvalues can differ significantly. These eigenvalues and their corresponding eigenvectors are adapted to the local image structure. A popular descriptor of the local image geometry is the structure tensor of Di Zenso [75]. In its most sophisticated form, it is given by the symmetric positive semidefinite matrix

$$\boldsymbol{J}_{n_c,\rho,\sigma} := K_\rho * \left(\sum_{k=1}^{n_c} \boldsymbol{\nabla}_\sigma\, u_k \boldsymbol{\nabla}_\sigma u_k^\top \right) \tag{4.5}$$

with eigenvalues $\mu_{1,\rho,\sigma} \geq \mu_{2,\rho,\sigma} \geq 0$. The corresponding diffusion tensor $\boldsymbol{D} := g(\boldsymbol{J}_{n_c,\rho,\sigma})$ uses the same set of eigenvectors and obtains its eigenvalues as functions of $\mu_{1,\rho,\sigma}$ and $\mu_{2,\rho,\sigma}$. The anisotropic models of Weickert/Brox [296] and Tschumperlé/Deriche [281] do not incorporate any smoothing in the structure tensor (i.e. $\sigma = \rho = 0$). However, such models degenerate to isotropic diffusion on greyscale images ($n_c = 1$). The methods of Roussos/Maragos [239] and Scharr et al. [246] involve a smoothing scale $\rho > 0$ and remain also anisotropic for $n_c = 1$. While Roussos/Maragos use $\sigma = 0$, Scharr et al. consider the case $\sigma > 0$ and replace all gradients $\boldsymbol{\nabla}$ by their Gaussian-smoothed counterparts $\boldsymbol{\nabla}_\sigma$. Tab. 4.1 in Section 4.5 provides a full description of the associated diffusion equations for each of these models.

Figure 4.1.: **Natural Images.** Two examples of natural images from the Berkeley database [188]. The database contains a large variety of different motives, e.g. animal photography, portraits, but also man made structures and machines.

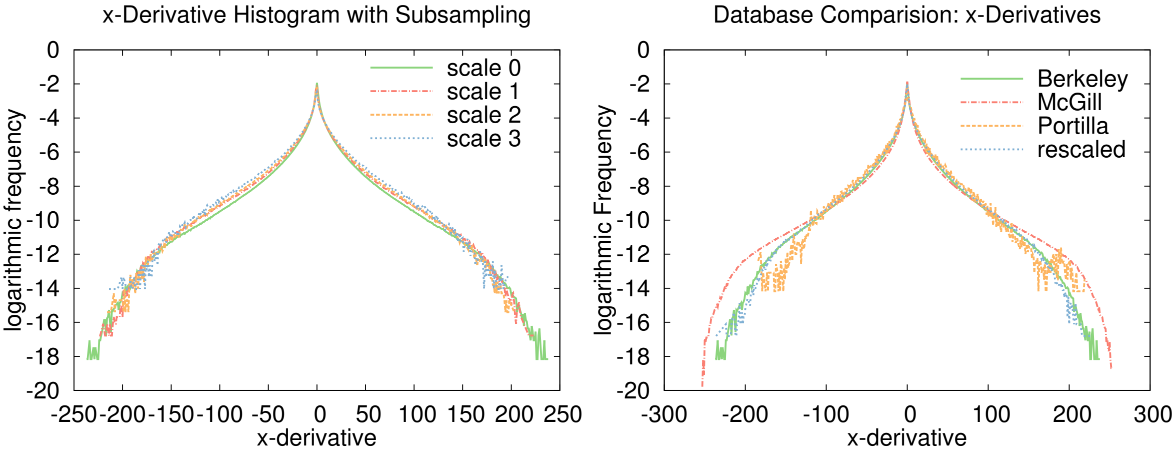
4.2 Probabilistic Foundations and Related Work

> *The probable is what usually happens.*
>
> — **Aristotle**
> (Philosopher)

Since we investigate the connections between probabilistic frameworks for image processing and diffusion-based image processing models, we briefly introduce some of the foundations needed to establish connections between both classes of methods. First, we give an overview over related work and continue to discuss the most important prior contributions in more detail in the rest of this section.

Overview of Related Work

At its core, our work relies on the non-Gaussian nature of the histograms that result from applying filters to so-called *natural images*. There is no universal definition for such images, but most works use databases containing photographs with a large variety of different motives such as the Berkeley database [188] (see Figure 4.1 for examples). The specific attributes of wavelet coefficients on image databases were first reported by Field [93]. These observations were systematically investigated for both derivative filters and wavelet coefficients by Huang and Mumford [134]. Invariances of these statistics are vital for their practical relevance. Zhu and Mumford [323] proposed that these statistical priors are invariant to scale and verified this empirically. In Figure 4.2 (a) we have reproduced these experiments on the Berkeley database [188]: Subsampling by averaging the pixels in 2×2 windows changes the histogram of derivatives in x-direction only marginally.



(a) scale invariance

(b) content invariance

Figure 4.2.: **Invariances of Derivative Statistics. (a):** Logarithmic frequency of occurrence for x-derivatives on the Berkeley database [188] over different scales. The i-th scale corresponds to i subsamplings of the images. **(b):** Comparison of logarithmic frequency of x-derivatives for the Berkeley database, a rescaled Berkeley database where each image has been randomly subsampled, the McGill [211] database and the Portilla test set [226].

Evaluations on databases containing different motives were conducted by Huang and Mumford [133]. While statistics can differ for databases with different homogeneous content (comparing e.g. a medical database with one containing purely animal photography), they are very consistent between natural image databases with a large variety of image quantity, sizes and content. Our experiments in Figure 4.2 (b) show the particularly remarkable similarity between the derivative priors of the large Berkeley database containing hundreds of images and the statistics of the Portilla test set [226], a collection of five well-known test images (e.g. *barbara* and *boats*).

The investigation of the non-Gaussian nature of natural image statistics is still an ongoing research topic, for example Pedersen et al. [214] considered probabilistic modelling with Lévy processes more recently. For more detailed discussions of the statistics of natural images than we can provide in this chapter, we refer to the survey paper of Srivastava et al. [263] and the recent monograph of Pouli et al. [227].

General connections between diffusion processes and statistical image processing models have been pioneered by Zhu and Mumford [323] within a Gibbs diffusion–reaction framework. Later on, Roth and Black [237, 238] have found additional relations in the context of Field-of-Experts (FoE). We discuss this particular kind of probabilistic model in more detail in the next section.

Works considering anisotropic diffusion models are, however, very rare. In the context of parameter learning, Scharr et al. [246] introduced an anisotropic model with Gaussian derivatives. A more recent parameter-free model goes back to Krajsek and Scharr [158]. They consider a two step procedure. In the first step, an isotropic diffusion process is derived. Afterwards, this is used to construct a *linear* anisotropic regularisation model. In a more systematic manner, Krajsek [157] deals with the connection between statistics and diffusion in his doctoral thesis, which is the closest work to our own. While he observes the relation from a primarily statistical point of view, we do the same with a focus on the diffusion perspective. It should also be mentioned that our probabilistic investigations lead to a framework that, on the pure diffusion side, has similarities to the one of Tschumperlé [282].

Beyond the context of direct relations between diffusion models and statistics, modelling and parameter choices for diffusion models have a long tradition in the literature. For isotropic diffusion, Perona and Malik [217] proposed to adapt the contrast parameter to quantiles of the image histogram. In medical imaging, Yoo and Coggins [317] learned diffusivities of regularised vector-valued isotropic diffusion according to a user-defined tissue classification prior. In a similar context, Beekman et al. [23] have investigated optimal parameter choices for 3-D nonlinear isotropic diffusion. They also compared to linear diffusion. More recently, Kunisch and Pock [159] proposed a bilevel optimisation scheme for parameter learning in the context of regularisation methods. Finally, Chen et al. [58] built nonlinear diffusion-reaction systems from learned linear filters and diffusivities. They apply them to image denoising and an image deblocking algorithm inspired by the FoE approach.

Statistical priors are also relevant for other areas of image processing. For example, Brox and Cremers [40] used Bayesian models for segmentation, namely by establishing the connection between image statistics and the Mumford-Shah functional [199]. Probabilistic Learning can also contribute to opening up new fields of application for PDEs: Saliency detection is a higher order vision task that deals with classifying image regions according to human attention. By learning systems of PDEs and their boundary conditions, Liu et al. [172] were able to build competitive saliency detectors.

Markov Random Field Models and Gibbs Distributions

In current literature on image processing with statistics, the object of interest, e.g. texture [324] or whole images of a certain class [238], are interpreted from the perspective of Markov random fields (MRFs). Such MRF approaches model each pixel as a distinct *random variable*. Consequently, the whole image is given by a random field which provides a probability measure of all possible configurations of grey values. In the following we provide definitions and conventions for MRFs that are convenient for our use in image processing. Note that there are also more ways to define different kinds of MRFs that rely on less or more restrictive assumptions.

In contrast to the diffusion models introduced in the previous section on a continuous image $u : \Omega \to \mathbb{R}$, we define MRFs on discrete images. For this purpose, we consider a vector $\boldsymbol{u} \in \mathbb{R}^{n_x n_y}$ where Ω is discretised by a rectangular grid with n_x pixels in x- and n_y pixels in n_y-direction using grid sizes h_x and h_y respectively. The vector notation is obtained by concatenating the image rows to a vector of size $n_x n_y$, thereby reducing the 2-D pixel index to a 1-D index. For more details on these index mappings see Chapter 2.3.

The corresponding MRF is a graph (V, E) where the set of vertices V represents the random variables associated to each component of $\boldsymbol{u}$. The undirected edges E describe conditional dependencies between the random variables. In our image processing case, the choice of these edges reflects the underlying neighbourhood dependencies of image processing filters. For example, a simple x-derivative that is approximated by a forward difference $u_x \approx (u_{i+1} - u_i)/h_x$ implies edges between all pixels in x-direction, but not in y-direction.

MRFs have to fulfil the *Markov property*: The conditional probability of any individual pixel given the remainder of the image can be reduced to the probability given the graph neighbours of the pixel. MRFs with strictly positive distributions are also referred to as Gibbs random fields (GRFs) and are of particular importance for our work due to a factorisation property that was originally proposed by Hammersley and Clifford [118] in an unpublished paper and later popularised by Besag [28].

Let $\mathcal{N} \subset \mathcal{P}(\{1, \dots, n_x n_y\})$ define a *neighbourhood system*. As a selection from all possible index sets contained in the power set $\mathcal{P}(\{1, \dots, n_x n_y\})$, it describes the neighbourhoods of pixels from $\boldsymbol{u}$. A set $C \in \mathcal{N}$ defines a *neighbourhood clique* $\boldsymbol{u}_C = \{u_i | i \in C\}$ if all vertices in C are pairwise connected by edges from E. In our case, the admissible clique configurations are determined by discrete filters that we want to apply to the image.

The *Hammersley-Clifford theorem* states that the probability density described by a GRF on a given neighbourhood system $\mathcal{N}$ can be factorised over the cliques in $\mathcal{N}$ in the form

$$p(\boldsymbol{u}) = \prod_{C \in \mathcal{N}} G_C(\boldsymbol{u}_C). \tag{4.6}$$

Thereby the probability density is expressed as a product of the so-called *potential functions* G_C evaluated at different vectors $\boldsymbol{u}_C \in \mathbb{R}^{|C|}$ which contain the components of $\boldsymbol{u}$ that belong to the respective clique C. In this thesis, we are only interested in *homogeneous* GRFs, where the potential G_C is identical for all cliques C, i.e. it is not dependent on the spatial location of the neighbourhood, just on the pixel values contained in the neighbourhood.

State of the art MRF methods like the FoE approach by Roth and Black [238] use this factorisation property to construct the spatially independent potential function G from a product of so-called *expert distributions* ϕ_k of filters $\boldsymbol{g}_k \in \mathbb{R}^{|C|}$, $k = 1, \dots, n_f$:

$$G(\boldsymbol{u}_C) := \prod_{k=1}^{n_f} \phi_k(\boldsymbol{g}_k^\top \boldsymbol{u}_C). \tag{4.7}$$

The filters $\boldsymbol{g}_k$ constitute a feature set that can be used to model the distribution of a given class of images. The FoE does not use predetermined filters, but tries to find a feature set that characterises the training data well. Interestingly, many of the filters learned by Roth and Black [238] resemble derivative filters. *Student-t experts* ϕ_k of the form given in Equation 4.8 reflect the non-Gaussian nature of the statistics of the filtering results. Each expert adapts to the individual distribution of a filter $\boldsymbol{g}_k$ with an appropriate choice of the free parameter $\alpha_k > 0$ in

$$\phi_k(\boldsymbol{g}_k^\top \boldsymbol{u}_C) := \left(1 + \frac{1}{2}(\boldsymbol{g}_k^\top \boldsymbol{u}_C)^2\right)^{-\alpha_k}. \tag{4.8}$$

Note that we have formulated the application of a linear filter to a clique by a dot product between a vector $\boldsymbol{g}_k \in \mathbb{R}^{|C|}$ containing the filter weights and the clique vector $\boldsymbol{u}_C$ so far. For our purposes in the rest of the chapter, it is more convenient to rewrite the clique-based notation of Roth and Black [238] into an equivalent pixel-based notation. We define the linear filter as a matrix $\boldsymbol{G}_k \in \mathbb{R}^{|\mathcal{N}| \times n_x n_y}$ instead. This allows us to write the application of the filter to the clique from $\mathcal{N}$ as an interaction with the whole image $\boldsymbol{u} \in \mathbb{R}^{n_x n_y}$: Each component of the vector $\boldsymbol{G}_k \boldsymbol{u}$ represents the result of the filtering of one clique. This is achieved by constructing the rows of the matrix from appropriately shifted entries of $\boldsymbol{g}_k$ and additional zero entries. Applying this notation to the Equations 4.6, 4.7 and 4.8, one obtains the FoE model

$$p(\boldsymbol{u}) = \prod_{i=1}^{n_x n_y} \prod_{k=1}^{n_f} \phi_k((\boldsymbol{G}_k \boldsymbol{u})_i). \tag{4.9}$$

Just as the previous section about diffusion, this brief summary just scratches the surface of MRF-based methods and only introduces the most important concepts that are relevant for the following sections. For further reading, we recommend the introduction to MRFs in image processing by Pérez [216].

Relating Isotropic Diffusion to Derivative Statistics

The connection between isotropic diffusion and MRF models is known, since Zhu and Mumford [323] have shown a correspondence between a prior learning model and a modification of the diffusion model defined by Equation 4.3. In the following, we reformulate this well-known connection as a direct correspondence between an MRF model and an isotropic diffusion equation. This formulation acts as a motivation and starting point for our further investigations on anisotropic models in the following sections.

First, consider a simplification of the Fields of Experts model [238] from the previous section. Instead of multiple different linear filters, we only use a single derivative filter $\boldsymbol{G}$ that approximates the squared gradient magnitude $|\boldsymbol{\nabla} u|^2$. This construction yields a simplified version of the distribution from Equation 4.9:

$$p(\boldsymbol{u}) = \prod_{i=1}^{n_x n_y} \phi((\boldsymbol{G}\boldsymbol{u})_i). \tag{4.10}$$

To illustrate the connection between the MRF model given above and diffusion models, we consider a Bayesian model for image denoising. The goal is to compute an approximation $\boldsymbol{u}$ to the original image $\boldsymbol{v}$ under two assumptions: $\boldsymbol{v}$ belongs to the class of natural images and is degraded by Gaussian noise. For any image $\boldsymbol{u}$, let $p(\boldsymbol{u})$ be the natural image prior. It describes the probability that $\boldsymbol{u}$ is a natural image and is derived from the statistics of image features on a suitable database. Furthermore, an assumption on the distribution of the noise yields the noise prior $p(\boldsymbol{f}|\boldsymbol{u})$. According to Bayes' rule, the posterior probability for a candidate image $\boldsymbol{u}$ to be the ground truth to an observed noisy image $\boldsymbol{f}$ obeys

$$p(\boldsymbol{u}|\boldsymbol{f}) \sim p(\boldsymbol{f}|\boldsymbol{u}) \cdot p(\boldsymbol{u}). \tag{4.11}$$

Thus, the denoised image $\hat{\boldsymbol{u}}$ can be obtained by maximising the posterior probability $p(\boldsymbol{u}|\boldsymbol{f})$ over all candidates $\boldsymbol{u}$:

$$\hat{\boldsymbol{u}} = \underset{\boldsymbol{u}}{\operatorname{argmax}}\, p(\boldsymbol{u}|\boldsymbol{f}). \tag{4.12}$$

In addition to the natural image prior from Eq. 4.10 we also need to define the noise prior $p(\boldsymbol{u}|\boldsymbol{f})$. A denoised image $\boldsymbol{u}$ should come close to the original $\boldsymbol{v}$ and thus the difference between $\boldsymbol{u}$ and the noisy image $\boldsymbol{f}$ should contain mostly the removed noise. Therefore, we can assume that this difference obeys the distribution of the noise. We assume Gaussian noise with standard deviation σ and mean zero in the following, which yields the noise prior

$$p(\boldsymbol{f}|\boldsymbol{u}) \sim \prod_{i=1}^{n_x n_y} \exp\left(-\frac{1}{2\sigma^2}(u_i - f_i)^2\right). \tag{4.13}$$

We can now combine both priors into a function $E_P(\boldsymbol{u})$ that we have to maximise according to Eq. 4.12:

$$E_P(\boldsymbol{u}) = \prod_{i=1}^{n_x n_y} \left(\exp\left(-\frac{1}{2\sigma^2}(u_i - f_i)^2\right) \cdot \phi((\boldsymbol{G}\boldsymbol{u})_i)\right). \tag{4.14}$$

In order to establish a connection to diffusion, we use the fact that this formulation of Bayesian denoising is remarkably similar to so-called *variational methods* for denoising. Instead of a discrete process, let us consider continuous counter parts $\hat{u}$, u, and f to the discrete denoising result, denoising candidate, and noisy image from Eq. 4.12. Variational methods aim to remove the noise by minimising the continuous energy functional

$$E(u) = \int_\Omega \left(\frac{1}{\tau} \underbrace{(u-f)^2}_{:=D(u,f)} + \underbrace{\psi(|\boldsymbol{\nabla} u|^2)}_{:=S(u)} \right) dx\, dy. \tag{4.15}$$

This formulation requires minimisation instead of the maximisation in the Bayesian approach, since it relies on penalisation: The *data term* $D(u, f)$ penalises deviations from the noisy image while the *smoothness term* $S(u)$ penalises deviations from the smoothness assumption given as a penaliser function ψ of the squared gradient magnitude. The parameter τ weights both assumptions.

In order to further investigate the connections between both denoising approaches, we can also reformulate the probabilistic method as a minimisation process. By employing a negative logarithm to Eq. 4.12, one obtains an equivalent energy minimisation process of the form

$$\hat{\boldsymbol{u}} = \underset{\boldsymbol{u}}{\operatorname{argmin}}\, E_{\log}(\boldsymbol{u}), \tag{4.16}$$

$$E_{\log}(\boldsymbol{u}) := -\log(p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})). \tag{4.17}$$

The logarithmically rescaled energy $E_{\log}$ offers an even closer resemblance to the continuous model:

$$E_{\log}(\boldsymbol{u}) = \sum_{i=1}^{n_x n_y} \left(\frac{(u_i - f_i)^2}{2\sigma^2} - \log \phi((\boldsymbol{G}\boldsymbol{u})_i) \right). \tag{4.18}$$

If we choose $\psi := -\log \phi$ and $\tau \sim 2\sigma^2$, E_{log} can be interpreted as a discretisation of the continuous energy functional E from Eq. 4.15. This connection acts as an intermediate step that allows us to reach our initial goal: connecting the Bayesian priors to diffusion. It is well known from variational calculus, that minimisers of the functional E necessarily fulfil the Euler-Lagrange equation

$$\frac{u-f}{\tau} = -\operatorname{div}\left(\psi'(|\boldsymbol{\nabla} u|^2)\boldsymbol{\nabla} u\right). \tag{4.19}$$

According to Scherzer and Weickert [248], the left-hand side of Equation 4.19 can be interpreted as a discretisation of $\partial_t u$ with one discrete time step of size τ. Thus, Eq. 4.19 is a time-discrete version of the *nonlinear isotropic* diffusion Equation 4.3.

So far, we have only reformulated the results of Zhu and Mumford [323] for a specific isotropic diffusion model. However, a closer investigation of this correspondence yields some insights that we can exploit in the following sections. The data term $D(u, f)$ assumes the role of the noise prior $p(\boldsymbol{u}|\boldsymbol{f})$ from the Bayesian model, while the smoothness term $S(u)$ is the counterpart of the natural image prior $p(\boldsymbol{u})$. Moreover, the diffusivity g from the PDE-formulation of the isotropic diffusion can be identified as the derivative ψ' of the negative logarithm of the potential ϕ.

In the following, we want to generalise these concepts: For a given diffusion model, we want to establish a connection to the histogram of a suitable discrete derivative filter. Note that the squared gradient magnitude which we have used as a natural image prior in this section is a purely isotropic image feature. Since we are interested in anisotropic diffusion models, we also need a direction-dependent statistical prior. To this end, we investigate the statistics of the structure tensor from Section 4.1.

4.3 Structure-adaptive Analysis of the Berkeley Database

> *By a small sample we may judge of the whole piece.*
>
> — **Miguel de Cervantes Saavedra**
>
> (Novelist and Poet)

Interpretation of the Structure Tensor. The local image structure of a vector-valued image $\boldsymbol{u}$ with n_c channels can be characterised by the joint structure tensor from Eq. 4.5. Its eigenvalues $\mu_1 \geq \mu_2$ represent the local contrast in the direction of the corresponding eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. For $\mu_1 \gg \mu_2$, the eigenvector $\boldsymbol{v}_2$ describes the direction of coherent structures while $\boldsymbol{v}_1$ points across these structures. Locally isotropic image content is characterised by $\mu_1 \approx \mu_2$. Thus, the eigenvalues of the structure tensor are image features that describe local geometry.

The Gaussian smoothing scales σ and ρ play distinct roles for the analysis of local image structure: Smoothing with K_σ removes noise and small-scale details. Thus, it should be chosen as small as possible. The smoothing scale ρ is usually chosen to be larger since its task is to accumulate neighbourhood information in the structure tensor.

In order to analyse the local geometry of a database, we need to discretise the structure tensor $\boldsymbol{J}_{n_c,\rho,\sigma}$ from Eq. 4.5. Since we are interested in anisotropy, rotation invariance is an important criterion for discretisations. To this end, we employ the LSAS scheme [305] for rotationally invariant tensor-driven diffusion processes. Let us consider a discrete multi-channel image $\boldsymbol{u}$ with $n_x \times n_y$ pixels and n_c channels. Each element $u_{k,i,j}$ describes the pixel value at the grid point (i,j) in channel k. The following finite differences form the building blocks for the discretisation on a 2×2-pixel neighbourhood $\{i,\, i{+}1\} \times \{j,\, j{+}1\}$ around a pixel $(i+\frac{1}{2}, j+\frac{1}{2})$:

$$\boxed{\substack{\\ \bullet\!\!-\!\!\bullet}} \;:=\; D_x^- u_{k,\,i+\frac{1}{2},\,j+\frac{1}{2}} \;:=\; \frac{u_{k,i+1,j} - u_{k,i,j}}{h_x}, \tag{4.20}$$

$$\boxed{\substack{\bullet\!\!-\!\!\bullet\\ }} \;:=\; D_x^+ u_{k,\,i+\frac{1}{2},\,j+\frac{1}{2}} \;:=\; \frac{u_{k,i+1,j+1} - u_{k,i,j+1}}{h_x}, \tag{4.21}$$

$$\begin{aligned} \boxed{\vert} \;&:=\; D_y^- u_{k,\,i+\frac{1}{2},\,j+\frac{1}{2}} \;:=\; \frac{u_{k,i,j+1} - u_{k,i,j}}{h_y}, && (4.22)\\ \boxed{\vert} \;&:=\; D_y^+ u_{k,\,i+\frac{1}{2},\,j+\frac{1}{2}} \;:=\; \frac{u_{k,i+1,j+1} - u_{k,i+1,j}}{h_y}, && (4.23) \end{aligned}$$

where h_x and h_y denote the grid sizes in the x- and y-direction. Welk et al. [305] propose the following discretisations for the components of the structure tensor $\boldsymbol{J}_k$ for channel k in pixel $(i + \frac{1}{2},\, j + \frac{1}{2})$:

$$\begin{aligned} J_{k,1,1} &= (\partial_x u_k)^2 &&\approx \frac{1}{4}\left(\left(\boxed{\overline{}} + \boxed{\underline{}}\right)^2 + \alpha \cdot \left(\boxed{\overline{}} - \boxed{\underline{}}\right)^2\right), && (4.24)\\ J_{k,2,2} &= (\partial_y u_k)^2 &&\approx \frac{1}{4}\left(\left(\boxed{\vert} + \boxed{\vert}\right)^2 + \alpha \cdot \left(\boxed{\vert} - \boxed{\vert}\right)^2\right), && (4.25)\\ J_{k,1,2} &= \partial_x u_k \, \partial_y u_k &&\approx \frac{1}{4}\left(\boxed{\overline{}}\,\boxed{\vert} + \boxed{\overline{}}\,\boxed{\vert} + \boxed{\underline{}}\,\boxed{\vert} + \boxed{\underline{}}\,\boxed{\vert}\right) && (4.26) \end{aligned}$$

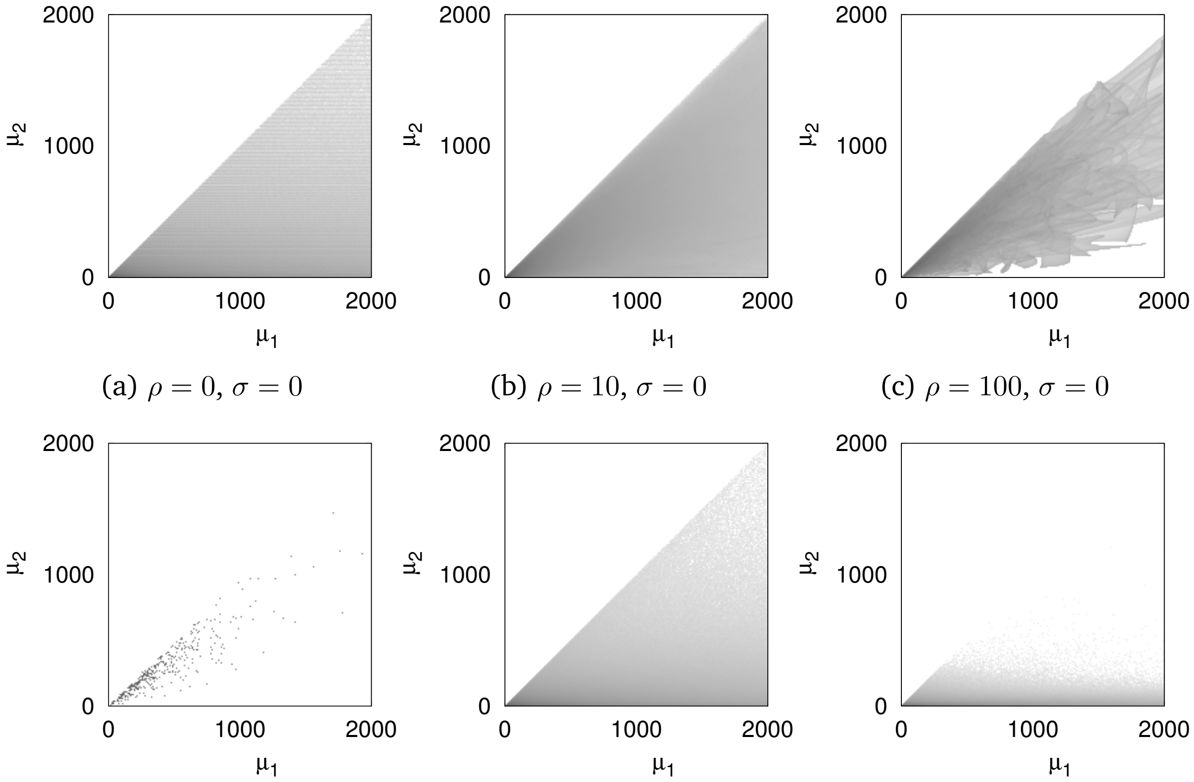
with a free parameter $\alpha \in [0, \frac{1}{2}]$. Note that this discretisation is a special case ($\beta = 0$) of the general framework by Weickert et al. [299] which we have discussed in Chapter 3.1. For $\alpha = 0$, one obtains the standard discretisation. For our purposes, we use $\alpha = 1/6$, a value that has been experimentally found to provide a good approximation of a rotationally invariant behaviour.

Anisotropic Statistics of Colour Images. Let us now use the aforementioned structure tensor for a statistical analysis of the Berkeley database [188]. The histogram of the eigenvalue pairs (μ_1, μ_2) with $\sigma = \rho = 0$ is displayed in Figure 4.3 (a). The fact that the eigenvalue μ_1 clearly dominates and that there are many structure tensors where μ_2 is significantly smaller confirms two things: Firstly, colour images contain many strongly oriented structures which legitimates the use of anisotropic filters. Secondly, these structures have some correlations over the colour channels. Figure 4.5 (a) reveals that both eigenvalues have the heavy-tailed distributions that are characteristic for filter results on natural images. Such kurtotic distributions are captured well by the function

$$\psi(x^2) = \frac{\lambda^2}{1-\gamma}\left(1 + \frac{x^2}{\lambda^2}\right)^{1-\gamma}. \tag{4.27}$$

The free parameters λ and γ can be adapted to fit ψ to the discrete histograms. A related model with one more degree of freedom was proposed by Krajsek and Scharr [158]. Similar statistics have been shown to be nearly identical on many databases of natural images such as the Berkeley [188] or McGill [211] test sets. In particular, they are also invariant for image content on different scales. Therefore, they form a good prior for natural images. This scale invariance implies that the statistics do hardly change under subsampling.

Invariances of the Structure Tensor. Of course, scale invariance remains an important property for our prior. Therefore, we investigate the behaviour of the histograms under subsampling in Figure 4.4. As for the x-derivatives in Figure 4.2 (a), the histograms change only marginally. Thus, the eigenvalues of the structure tensor fulfil the important scale



(a) $\rho = 0$, $\sigma = 0$

(b) $\rho = 10$, $\sigma = 0$

(c) $\rho = 100$, $\sigma = 0$

(d) $\rho = 10^5$, $\sigma = 0$

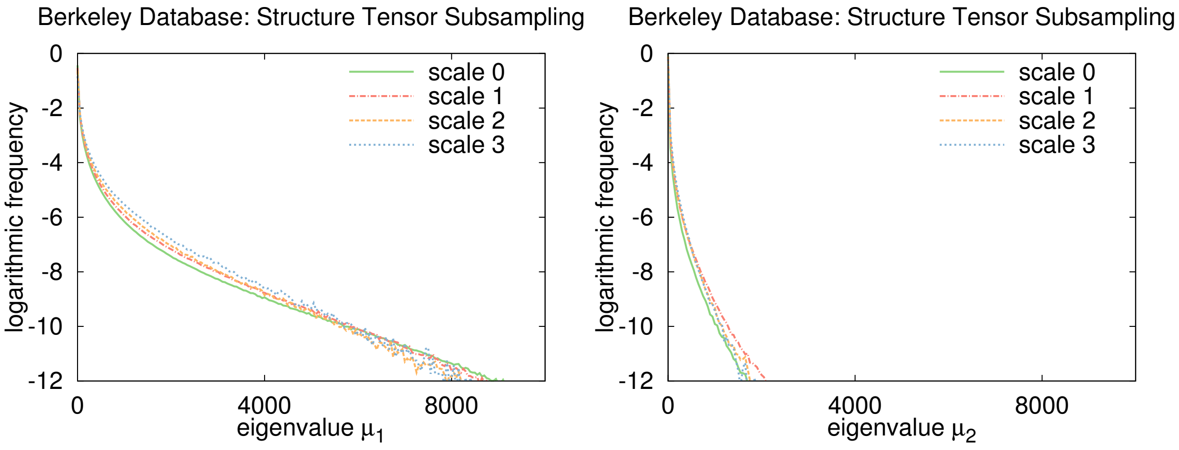
(e) $\rho = 0$, $\sigma = 0.5$

(f) $\rho = 0$, $\sigma = 1$

Figure 4.3.: **Eigenvalue histogram of the structure tensor on the Berkeley image database.** The graph shows the evolution of the negative logarithmic histogram of the eigenvalue pairs $(\mu_{1,\rho,\sigma}, \mu_{2,\rho,\sigma})$ of the structure tensor $\boldsymbol{J}_{n_c,\rho,\sigma}$ over different scales ρ and σ. Dark values indicate high occurrences and bright values low occurrences. Values on the diagonal indicate isotropic tensors, the amount of deviation from the diagonal stands for increasing anisotropy.

invariance property. Note that subsampling is a form of averaging: Neighbourhoods of several pixels are replaced by their weighted average in each image and the statistics are computed afterwards. The structure tensor itself also inherently contains averaging in the form of the Gaussian convolutions with smoothing scales σ and ρ. However, in contrast to subsampling, the Gaussian smoothing kernels overlap. In the following we investigate how this different kind of averaging affects the eigenvalue histograms.

Behaviour under Smoothing. If one averages with overlapping neighbourhoods, the statistics depend significantly on the neighbourhood size. This happens for the Gaussian-smoothed structure tensor $\boldsymbol{J}_{n_c,\rho,\sigma}$, where the tensor entries are embedded in a Gaussian scale-space. Let us first fix σ and consider the scale-space behaviour with respect to ρ. Figure 4.3 (a)–(d) shows the evolution of the histogram for the eigenvalue pairs $(\mu_{1,\rho,\sigma},\ \mu_{2,\rho,\sigma})$. We observe that for increasing ρ, the joint histogram clusters towards the diagonal. This shows that $\mu_{1,\rho,\sigma}$ and $\mu_{2,\rho,\sigma}$ approach each other, i.e. the structure tensor becomes more isotropic. This is plausible, since one smoothes over structures with different orientations. For $\rho \to \infty$, all tensors $\boldsymbol{J}_{n_c,\rho,\sigma}$ converge to the average structure tensor of the whole image. If all directions were equally



(a) scale invariance of μ_1

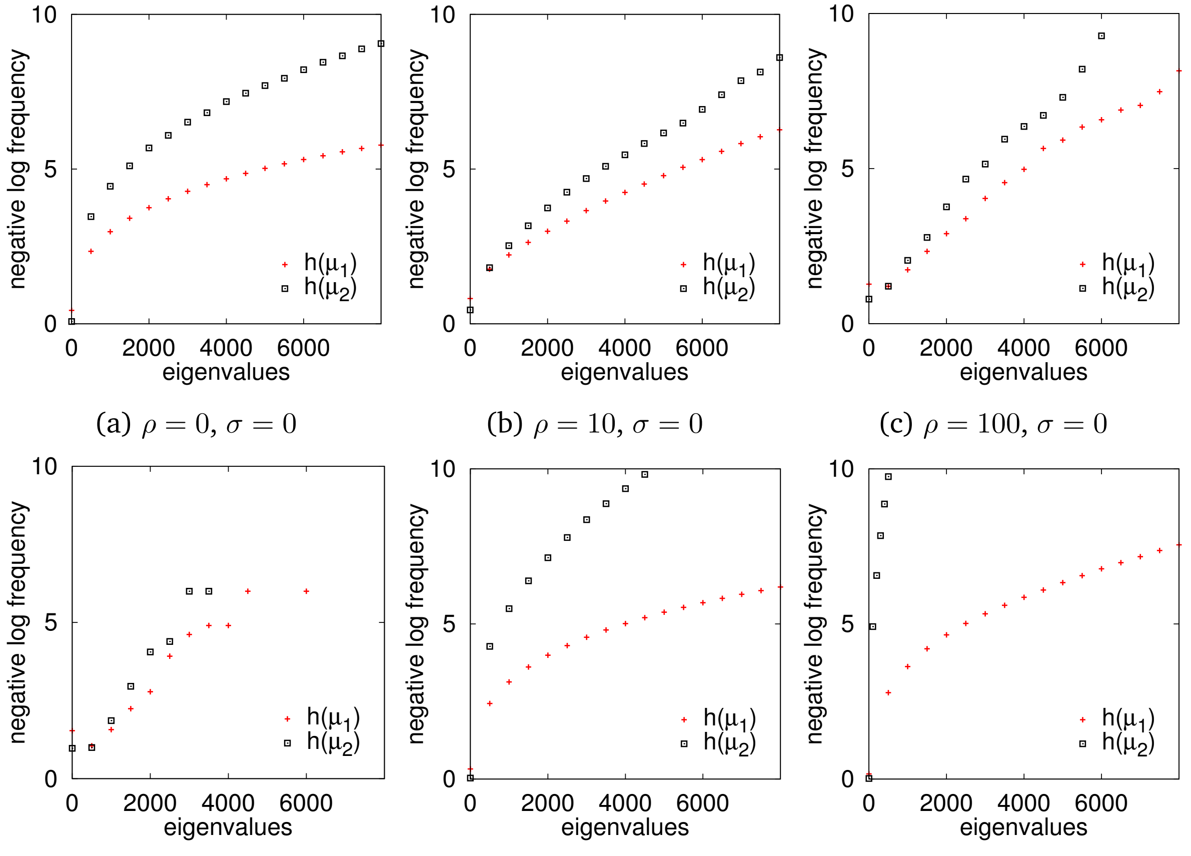
(b) scale invariance of μ_2

Figure 4.4.: **Scale-Invariance of the Structure Tensor.** The histograms of the structure tensor's eigenvalues μ_1 and μ_2 change only marginally, if subsampling is applied to the Berkeley database. Also note that the second eigenvalue μ_2 has a much more limited range than μ_1.

prominent over the database, this average tensor would be purely isotropic . However, the steady state of the statistics ($\rho = 10^5$ in Figure 4.3 (d) and Figure 4.5 (d)) reveals some anisotropy. Thus, the average eigenvalue histograms show the inherent directional bias of the image database.

Now we fix ρ and investigate the evolution under σ. For $\sigma \to \infty$, the local contrast given by $\mu_{1,\rho,\sigma}$ and $\mu_{2,\rho,\sigma}$ approaches 0 and the corresponding diffusion tensor $\boldsymbol{D}$ converges to the unit matrix $\boldsymbol{I}$. Interestingly, Figures 4.3 (e)–(f) and 4.5 (e)–(f) show that for small σ, the presmoothing increases the difference between the histograms of $\mu_{1,\rho,\sigma}$ and $\mu_{2,\rho,\sigma}$. This fosters anisotropy of the image prior. We conjecture that Gaussian convolution effectively removes high-frequent isotropic perturbations, such that anisotropic image structures become more dominant. For larger σ their dominance decreases again.

In conclusion, we observe that natural images contain pronounced anisotropies and their statistics strongly depend on the smoothing scales ρ and σ. This suggests to design filters that take into account such anisotropic phenomena as priors.



(a) $\rho = 0$, $\sigma = 0$ (b) $\rho = 10$, $\sigma = 0$ (c) $\rho = 100$, $\sigma = 0$

(d) $\rho = 10^5$, $\sigma = 0$ (e) $\rho = 0$, $\sigma = 0.5$ (f) $\rho = 0$, $\sigma = 1$

Figure 4.5.: Evolution of the negative logarithmic histograms $h(\mu_1)$, $h(\mu_2)$ of the eigenvalues $\mu_{1,\rho,\sigma}$, $\mu_{2,\rho,\sigma}$ of the structure tensor $\boldsymbol{J}_{n_c,\rho,\sigma}$ over different scales ρ and σ.

4.4 Probabilistic Denoising with a Structure Tensor Prior

> *Noise is a parasite. Anything noisy is poorly designed.*
>
> — **Raymond Loewy**
>
> (Designer)

After establishing and analysing the structure tensor as an anisotropic image prior, we now use the statistics from Section 4.3 as a prior for Bayesian denoising. As in Section 4.2, we obtain the denoised image $\hat{\boldsymbol{u}}$ by maximising the posterior probability $p(\boldsymbol{u}|\boldsymbol{f})$ over the candidates $\boldsymbol{u}$ given the image prior $p(\boldsymbol{u})$ and the noise prior $p(\boldsymbol{f}|\boldsymbol{u})$:

$$\hat{\boldsymbol{u}} = \underset{\boldsymbol{u}}{\operatorname{argmax}}\, p(\boldsymbol{u}|\boldsymbol{f}), \tag{4.28}$$

$$p(\boldsymbol{u}|\boldsymbol{f}) \sim p(\boldsymbol{f}|\boldsymbol{u}) \cdot p(\boldsymbol{u}). \tag{4.29}$$

Note however that this time, we consider multi-channel images of size $n_x \times n_y$ with n_c channels. Therefore, we also have to adapt the noise prior. We assume independent identically distributed Gaussian noise for each channel k with $k \in \{1, ..., n_c\}$, which yields

$$p(\boldsymbol{f}|\boldsymbol{u}) \sim \prod_{k=1}^{n_c} \prod_{i=1}^{n_x} \prod_{j=1}^{n_y} \exp\Big(-\frac{1}{2\sigma^2}(u_{k,i,j} - f_{k,i,j})^2\Big). \tag{4.30}$$

In order to formulate a natural image prior, we again follow the minimax entropy model that has been used to model texture [324] and whole images [323]. For a set of given linear or nonlinear filters $\{\boldsymbol{F}_1, \ldots, \boldsymbol{F}_{n_f}\}$ the distribution of natural images is modelled as

$$p(\boldsymbol{u}) = \prod_{\ell=1}^{n_f} \prod_{i=1}^{n_x} \prod_{j=1}^{n_y} \phi_\ell(\boldsymbol{F}_\ell(\boldsymbol{u})_{i,j}). \tag{4.31}$$

Here the potential functions ϕ_ℓ model the distribution of the corresponding filter $\boldsymbol{F}_\ell$. We need a more general formulation here than in Section 4.2 in order to capture multiple fundamentally different diffusion models.

Let $\phi(\mu_1, \mu_2)$ define the distribution of an arbitrary image feature that is derived from the eigenvalues μ_1 and μ_2 of $\boldsymbol{J}_{n_c,\rho,\sigma}$. In particular, this formulation also includes separate statistics for both eigenvalues, i.e. $\phi(\mu_1, \mu_2) := \phi_1(\mu_1) \cdot \phi_2(\mu_2)$. Such image features can be interpreted as *second-level priors* in the terminology of Zhu and Mumford [323], since they model the local geometry of image structures. In particular, these priors adapt to dominant directions in the image in contrast to linear filters that approximate derivatives in a fixed, global direction. By specifying the natural image prior (4.31) with a feature based on μ_1 and μ_2 and including the noise prior (4.30) we obtain the following energy:

$$\boxed{E_P(\boldsymbol{u}) = \prod_{i=1}^{n_x} \prod_{j=1}^{n_y} \left(\prod_{k=1}^{n_c} \exp\Big(-\frac{(u_{k,i,j} - f_{k,i,j})^2}{2\sigma^2}\Big)\right) \cdot \phi(\mu_{1,i,j}, \mu_{2,i,j}).} \tag{4.32}$$

Maximising E_P gives the denoised image $\hat{\boldsymbol{u}}$.

4.5 The Unifying Prior-Based Diffusion Framework

> *In the particular is contained the universal.*
>
> — **James Joyce**
>
> (Novelist)

Let us now show that the probabilistic denoising model (4.32) is the discrete counterpart to a unifying diffusion framework that incorporates a large family of existing diffusion approaches. Instead of maximising the energy E_P, we consider the minimisation of its negative logarithm

$$E_{\log}(\boldsymbol{u}) := \frac{1}{2}\sum_{i=1}^{n_x}\sum_{j=1}^{n_y}\left(\sum_{k=1}^{n_c}\frac{1}{\tau}(u_{k,i,j} - f_{k,i,j})^2 + \psi(\mu_{1,i,j},\, \mu_{2,i,j})\right) . \tag{4.33}$$

Here, we define the penaliser ψ as $\psi(\mu_1, \mu_2) = -\log\phi(\mu_1, \mu_2)$, and we choose $\tau \sim \sigma^2$. A variational regularisation approach is obtained by the minimisation of the continuous counterpart to $E_{\log}$:

$$\boxed{E(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left(\frac{1}{\tau}|\boldsymbol{u} - \boldsymbol{f}|^2 + \psi(\mu_1, \mu_2)\right) dx\, dy} \tag{4.34}$$

where $|\cdot|$ denotes the Euclidean norm. For scalar-valued images, this energy is identical to the one proposed by Scharr et al. [246]. Interestingly, this energy provides a unifying framework for the eight diffusion models from Section 4.1. The key result for understanding this connection is given by the following proposition.

Proposition 4.5.1. [Euler-Lagrange Equations of the General Energy Functional]

The energy functional $E(\boldsymbol{u})$ from Eq. (4.34) *gives rise to the Euler–Lagrange equations*

$$\frac{u_k - f_k}{\tau} = \boldsymbol{\nabla}_\sigma^\top\left((K_\rho * \boldsymbol{D})\ \boldsymbol{\nabla}_\sigma u_k\right), \qquad k = 1, ..., n_c, \tag{4.35}$$

with natural boundary conditions $\boldsymbol{n}^\top(K_\sigma * K_\rho * \boldsymbol{D}\boldsymbol{\nabla}_\sigma u_k) = 0$ on $\partial\Omega$. Here, $\boldsymbol{n}$ is the outer image normal and $\boldsymbol{D}$ is given in terms of the eigenvectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ and eigenvalues μ_1, μ_2 of the structure tensor $\boldsymbol{J}_{n_c,\sigma,\rho}$:

$$\boldsymbol{D} := \frac{\partial\psi(\mu_1, \mu_2)}{\partial\mu_1}\,\boldsymbol{v}_1\boldsymbol{v}_1^\top + \frac{\partial\psi(\mu_1, \mu_2)}{\partial\mu_2}\,\boldsymbol{v}_2\boldsymbol{v}_2^\top . \tag{4.36}$$

Note that the findings of Proposition 4.5.1. differ from the results of Scharr et al. [246].

Proof. The Euler-Lagrange equations are obtained from the Gâteaux derivatives of $E(\boldsymbol{u})$. We focus on the derivative of the penaliser ψ. With $d_{\epsilon_k}(f) := \frac{\partial}{\partial\epsilon_k} f|_{\epsilon_k=0}$, $k \in \{1, ..., n_c\}$, a test

Model	n_c	σ	ρ	PDE	Penaliser	Prior
Homogeneous Diffusion [141]	1	0	0	$\partial_t u = \Delta u$	$\psi(\mu_1+\mu_2) = \lvert\boldsymbol{\nabla} u\rvert^2$	$\phi(\mu_1+\mu_2) = e^{-\lvert\boldsymbol{\nabla} u\rvert^2}$
Perona/ Malik [217]	1	0	0	$\partial_t u = \boldsymbol{\nabla}^\top \Big(\psi'(\lvert\boldsymbol{\nabla} u\rvert^2\Big)\boldsymbol{\nabla} u)$	$\psi(\mu_1+\mu_2) = -\ln\phi(\lvert\boldsymbol{\nabla} u\rvert^2)$	$\phi(\mu_1+\mu_2) = \phi(\lvert\boldsymbol{\nabla} u\rvert^2)$
Gerig et al. [104]	≥ 1	0	0	$\partial_t u_k = \boldsymbol{\nabla}^\top(\psi'(\sum_{\ell=1}^{n_c}\lvert\boldsymbol{\nabla} u_\ell\rvert^2)\boldsymbol{\nabla} u_k)$	$\psi(\mu_1+\mu_2) = -\ln\phi\left(\sum\limits_{\ell=1}^{n_c}\lvert\boldsymbol{\nabla} u_\ell\rvert^2\right)$	$\phi(\mu_1+\mu_2) = \phi\left(\sum\limits_{\ell=1}^{n_c}\lvert\boldsymbol{\nabla} u_\ell\rvert^2\right)$
Scherzer/ Weickert [248]	1	≥ 0	0	$\partial_t u = \boldsymbol{\nabla}_\sigma^\top\Big(\psi'(\lvert\boldsymbol{\nabla}_\sigma u\rvert^2)\boldsymbol{\nabla}_\sigma u\Big)$	$\psi(\mu_1+\mu_2) = -\ln\phi(\lvert\boldsymbol{\nabla}_\sigma u\rvert^2)$	$\phi(\mu_1+\mu_2) = \phi(\lvert\boldsymbol{\nabla}_\sigma u\rvert^2)$
Weickert/ Brox [296]	> 1	0	0	$\partial_t u_k = \boldsymbol{\nabla}^\top\Big(\Big(\psi'(\mu_1)\boldsymbol{v}_1\boldsymbol{v}_1^\top + \psi'(\mu_2)\boldsymbol{v}_2\boldsymbol{v}_2^\top\Big)\boldsymbol{\nabla} u_k\Big)$	$\psi(\mu_{1,2}) = -\ln\phi(\mu_{1,2})$	$\phi(\mu_1)\cdot\phi(\mu_2)$
Tschumperlé/ Deriche [281]	> 1	0	0	$\partial_t u_k = \boldsymbol{\nabla}^\top\Big(\Big(\frac{\partial\psi(\mu_1,\mu_2)}{\partial\mu_1}\boldsymbol{v}_1\boldsymbol{v}_1^\top + \frac{\partial\psi(\mu_1,\mu_2)}{\partial\mu_2}\boldsymbol{v}_2\boldsymbol{v}_2^\top\Big)\boldsymbol{\nabla} u_k\Big)$	$\psi(\mu_1,\mu_2) = -\ln\phi(\mu_1,\mu_2)$	$\phi(\mu_1,\mu_2)$
Roussos/ Maragos [239]	≥ 1	0	> 0	$\partial_t u_k = \boldsymbol{\nabla}^\top\Big(K_\rho * \big(\frac{\partial\psi(\mu_1,\mu_2)}{\partial\mu_1}\boldsymbol{v}_1\boldsymbol{v}_1^\top + \frac{\partial\psi(\mu_1,\mu_2)}{\partial\mu_2}\boldsymbol{v}_2\boldsymbol{v}_2^\top\big)\boldsymbol{\nabla} u_k\Big)$	$\psi(\mu_1,\mu_2) = -\ln\phi(\mu_1,\mu_2)$	$\phi(\mu_1,\mu_2)$
Scharr et al. [246]	1	≥ 0	> 0	$\partial_t u = \boldsymbol{\nabla}_\sigma^\top\Big(K_\rho * \Big(\psi_1'(\mu_1)\boldsymbol{v}_1\boldsymbol{v}_1^\top + \psi_2'(\mu_2)\boldsymbol{v}_2\boldsymbol{v}_2^\top\Big)\boldsymbol{\nabla}_\sigma u\Big)$	$\psi_{1,2}(\mu_{1,2}) = -\ln\phi_{1,2}(\mu_{1,2})$	$\phi_1(\mu_1)\cdot\phi_2(\mu_2)$

Table 4.1.: **Existing diffusion models and their relation to the unifying framework.** The models primarily differ in the number of image channels n_c, the smoothing scale ρ for the structure tensor and the presmoothing scale σ. Additionally, most models impose certain restrictions to the general prior $\phi(\mu_1, \mu_2)$. The priors and penalisers are always given in the most general form that the corresponding model allows. Note that the Roussos/Maragos model has the same prior structure as Tschumperlé/Deriche, but yields a different PDE due to the nonzero smoothing scale ρ.

function $\boldsymbol{h} : \mathbb{R}^2 \mapsto \mathbb{R}^{n_c}$, the diagonal matrix $\boldsymbol{H} := \mathbf{diag}(\boldsymbol{h})$ (see Eq. G.3), and $\boldsymbol{\epsilon} \in \mathbb{R}^{n_c}$ we calculate:

$$d_{\epsilon_k}(\psi(\mu_1(\boldsymbol{u}+\boldsymbol{H}\boldsymbol{\epsilon}),\ \mu_2(\boldsymbol{u}+\boldsymbol{H}\boldsymbol{\epsilon}))) \ = \ \frac{\partial\psi}{\partial\mu_1} d_{\epsilon_k}(\mu_1) \ + \ \frac{\partial\psi}{\partial\mu_2} d_{\epsilon_k}(\mu_2). \tag{4.37}$$

Therefore, the derivatives of the eigenvalues μ_1 and μ_2 of $\boldsymbol{J}_{n_c,\rho,\sigma}$ must be computed. In terms of the matrix elements $J_{1,1}$, $J_{1,2}$, $J_{2,2}$, the eigenvalue μ_1 is given by

$$\mu_1 = \frac{1}{2}\left(J_{1,1} + J_{2,2} + \sqrt{(J_{1,1} - J_{2,2})^2 + 4J_{1,2}^2} \right). \tag{4.38}$$

By writing the derivatives $d_{\epsilon_k}(J_{1,2})$, $d_{\epsilon_k}(J_{1,1} + J_{2,2})$, and $d_{\epsilon_k}(J_{1,1} - J_{2,2})$ as dot products with $\boldsymbol{\nabla}_\sigma h_k$, we can simplify $d_{\epsilon_k}(\mu_1)$ to

$$d_{\epsilon_k}(\mu_1) \ = \ K_\rho * \left((\boldsymbol{M}\boldsymbol{\nabla}_\sigma u_k)^\top \boldsymbol{\nabla}_\sigma h_k \right), \tag{4.39}$$

$$\boldsymbol{M} \ := \ \frac{2}{\mu_1 - \mu_2} \begin{pmatrix} \mu_1 - \mu_2 + J_{1,1} - J_{2,2} & 2J_{1,2} \\ 2J_{1,2} & \mu_1 - \mu_2 - J_{1,1} + J_{2,2} \end{pmatrix}. \tag{4.40}$$

Algebraic computations similar to [239] lead to $\boldsymbol{M} = 2\boldsymbol{v}_1\boldsymbol{v}_1^\top$. With analogous results for $d_{\epsilon_k}(\mu_2)$, we obtain $d_{\epsilon_k}(\psi(\mu_1,\mu_2)) = (K_\rho * \boldsymbol{D}\boldsymbol{\nabla}_\sigma u_k)^\top \boldsymbol{\nabla}_\sigma h_k$. Plugging these results into the Gâteaux derivative $d_{\epsilon_k}E$ of the energy and applying partial integration yields

$$d_{\epsilon_k}E = \sum_{\ell=1}^{2} \left[\left(K_\sigma * K_\rho * \boldsymbol{D}\boldsymbol{\nabla}_\sigma u_k \right)_\ell h_k \right]_{a_\ell}^{b_\ell} - \int_\Omega \operatorname{div}_\sigma\left((K_\rho * \boldsymbol{D})\boldsymbol{\nabla}_\sigma u_k \right) h_k \, dx \, dy \tag{4.41}$$

with $\Omega = [a_1, b_1] \times [a_2, b_2]$. Variational calculus yields Eq. (4.35) and the natural boundary conditions $\boldsymbol{n}^\top(K_\sigma * K_\rho * \boldsymbol{D}\boldsymbol{\nabla}_\sigma u_k) = 0$ on $\partial\Omega$. In this proof sketch we have skipped many of the technicalities. Appendix G contains more details and intermediate steps. □

According to Scherzer and Weickert [248], Eq. (4.35) can be interpreted as an implicit time discretisation with one time step of size τ of the general diffusion equation

$$\boxed{\partial_t u_k \ = \ \boldsymbol{\nabla}_\sigma^\top \left((K_\rho * \boldsymbol{D}) \ \boldsymbol{\nabla}_\sigma u_k \right), \quad k = 1, ..., n_c} \tag{4.42}$$

with initial condition $\boldsymbol{u}(t = 0) = \boldsymbol{f}$. In Table 4.1 we demonstrate that a large number of existing diffusion models can be considered as special cases of this unifying partial differential equation. To see this, note that the isotropic models use $\rho = 0$ and the prior

$$\phi(\mu_1 + \mu_2) \ = \ \phi\left(\operatorname{tr}\boldsymbol{J}_{n_c,0,\sigma}\right) \ = \ \phi\left(\sum_{\ell=1}^{n_c} |\boldsymbol{\nabla}_\sigma u_\ell|^2 \right). \tag{4.43}$$

Moreover, for greyscale images ($n_c = 1$) and smoothing scale $\rho = 0$, the structure tensor $\boldsymbol{J}_{1,0,\sigma} = \boldsymbol{\nabla}_\sigma u \boldsymbol{\nabla}_\sigma u^\top$ has the normalised eigenvectors $\boldsymbol{v}_1 = \frac{\boldsymbol{\nabla}_\sigma u}{|\boldsymbol{\nabla}_\sigma u|}$ and $\boldsymbol{v}_2 = \boldsymbol{v}_1^\perp$. As a consequence, the diffusion process from Eq. (4.42) degenerates to isotropic diffusion with a scalar diffusivity: Using Eq. 4.36 we get

$$\begin{aligned} \boldsymbol{D}\,\boldsymbol{\nabla}_\sigma u &= \left(\frac{\partial\psi}{\partial\mu_1}\frac{\boldsymbol{\nabla}_\sigma u \boldsymbol{\nabla}_\sigma u^\top}{|\boldsymbol{\nabla}_\sigma u|^2} + \frac{\partial\psi}{\partial\mu_2}\frac{\boldsymbol{\nabla}_\sigma u^\perp \boldsymbol{\nabla}_\sigma u^{\perp\top}}{|\boldsymbol{\nabla}_\sigma u|^2}\right) \boldsymbol{\nabla}_\sigma u \\ &= \frac{\partial\psi}{\partial\mu_1}\,\boldsymbol{\nabla}_\sigma u \;=\; \psi'\left(|\boldsymbol{\nabla}_\sigma u|^2\right)\,\boldsymbol{\nabla}_\sigma u. \end{aligned} \tag{4.44}$$

Homogeneous diffusion is also captured by the model (4.42), if one chooses $\phi(|\boldsymbol{\nabla} u|^2) := \exp(-|\boldsymbol{\nabla} u|^2)$ as prior distribution. The four anisotropic models are covered as follows: Weickert/Brox [296] and Scharr et al. [246] use the factorised prior $\phi_1(\mu_1) \cdot \phi_2(\mu_2)$, in the case of Weickert/Brox with identical functions ϕ_1 and ϕ_2 and $\sigma = \rho = 0$. The models of Tschumperlé/Deriche [281] and Roussos/Maragos [239] allow general priors $\phi(\mu_1, \mu_2)$, but specify $\sigma := 0$. Moreover, Tschumperlé/Deriche also set $\rho := 0$.

The whole framework was derived from a common natural image prior, the directional statistics of the structure tensor. This shows that the ad hoc choices that were made for diffusion models during decades of research in fact reflect inherent properties of natural images. This observation can be even extended to the choice of diffusivities: If we consider the special case $\phi(\mu_1, \mu_2) = \phi_1(\mu_1) \cdot \phi_2(\mu_2)$, we are able to decompose $\psi(\mu_1, \mu_2) := \psi_1(\mu_1) + \psi_2(\mu_2)$ into two separate penalisers $\psi_\ell = -\ln \phi_\ell$ with $\ell \in \{1, 2\}$. The kurtotic distribution model (4.27) gives rise to the following family of diffusivities:

$$\psi'(x^2) \;=\; \left(1 + \frac{x^2}{\lambda^2}\right)^{-\gamma}. \tag{4.45}$$

Comparing this to Eq. 4.4 shows that the Perona/Malik diffusivity [217] is covered for $\gamma = 1$ and the Charbonnier diffusivity [54] results for $\gamma = 0.5$. To the best of our knowledge, our framework covers all relevant diffusion models that offer a variational interpretation. Since it is a variational framework, it is natural that it cannot be applied to models for which no variational formulation is known, e.g. edge- and coherence-enhancing diffusion filters [293].

4.6 Denoising Experiments

> *All life is an experiment.*
> *The more experiments you make, the better.*
>
> — **Ralph W. Emerson**
> (Poet)

In the following, we compare the performance of different diffusion models in the context of image denoising. We focus on those models from Table 4.1 that are designed for colour images and apply small modifications where necessary: In analogy to Gerig et al. [104], we extend the Scherzer/Weickert model to colour images by coupling the gradient within a joint diffusivity. Furthermore, we use the separate penalisers $\psi_1(\mu_1)$ and $\psi_2(\mu_2)$ from the model of Scharr et al. [246] for all anisotropic models. This extends the Weickert/Brox model with individual diffusivities for both eigenvalues, which is a special case of the Tschumperlé/Deriche model. In the accompanying figures we use the abbreviations H for homogeneous diffusion [141], GKKJ for Gerig et al. [104], SW for Scherzer/Weickert [248], WBTD for the hybrid model of Weickert/Brox [296] and Tschumperlé/Deriche [281], RM for Roussos/Maragos [239], and SBH for a vector-valued extension of Scharr et al. [246].

For our experiments, we first determine the parameters λ and γ of the prior distribution (4.27) and the corresponding diffusivity (4.45). To this end, we compute the discrete histograms of μ_1 and μ_2 on the 200 training images of the Berkeley database [188]. For a nonlinear least squares fit to these histograms, we have chosen the Matlab implementation of the Levenberg–Marquardt algorithm (version 3.2.1 of the Matlab curve fitting toolbox). In Figure 4.6 we see that the resulting diffusivities decrease more rapidly for μ_1 than for μ_2. Thus, they inhibit diffusion across coherent structures more than along them. For increasing smoothing scales σ and ρ this anisotropic behaviour is reduced, since the difference between the diffusivities ψ_1' and ψ_2' is less pronounced. In the following, we use $\rho = 0.5$ and $\sigma = 0.2$.

For our denoising experiments, we consider the partial differential formulation of the statistically-derived diffusion filters and apply them to the 100 images of the Berkeley test set [188] with added Gaussian noise. The average peak signal to noise (PSNR) values for different standard deviations of the noise are given in Tab. 4.2. We observe that for all noise levels, homogeneous diffusion H yields the worst results, and the isotropic methods GKKJ and SW perform consistently below the anisotropic models WBTD, RM and SBH. With increasing noise levels, the Gaussian smoothing scales σ and ρ within the models SW, RM and SBH offer a slight PSNR advantage over their counterparts GKKJ and WBTD that have to cope without Gaussian smoothing. Visually, the most distinct difference is the severe blurring of edges in homogeneous diffusion that sets it apart from the other models.

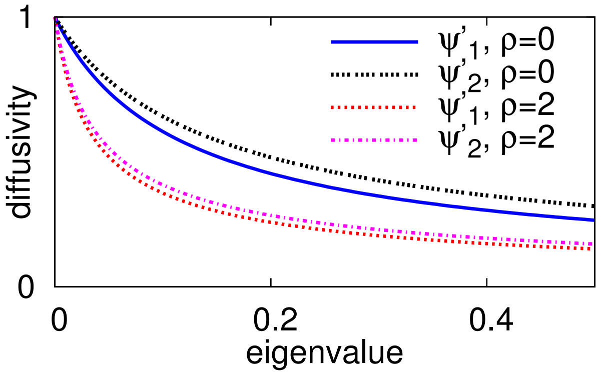
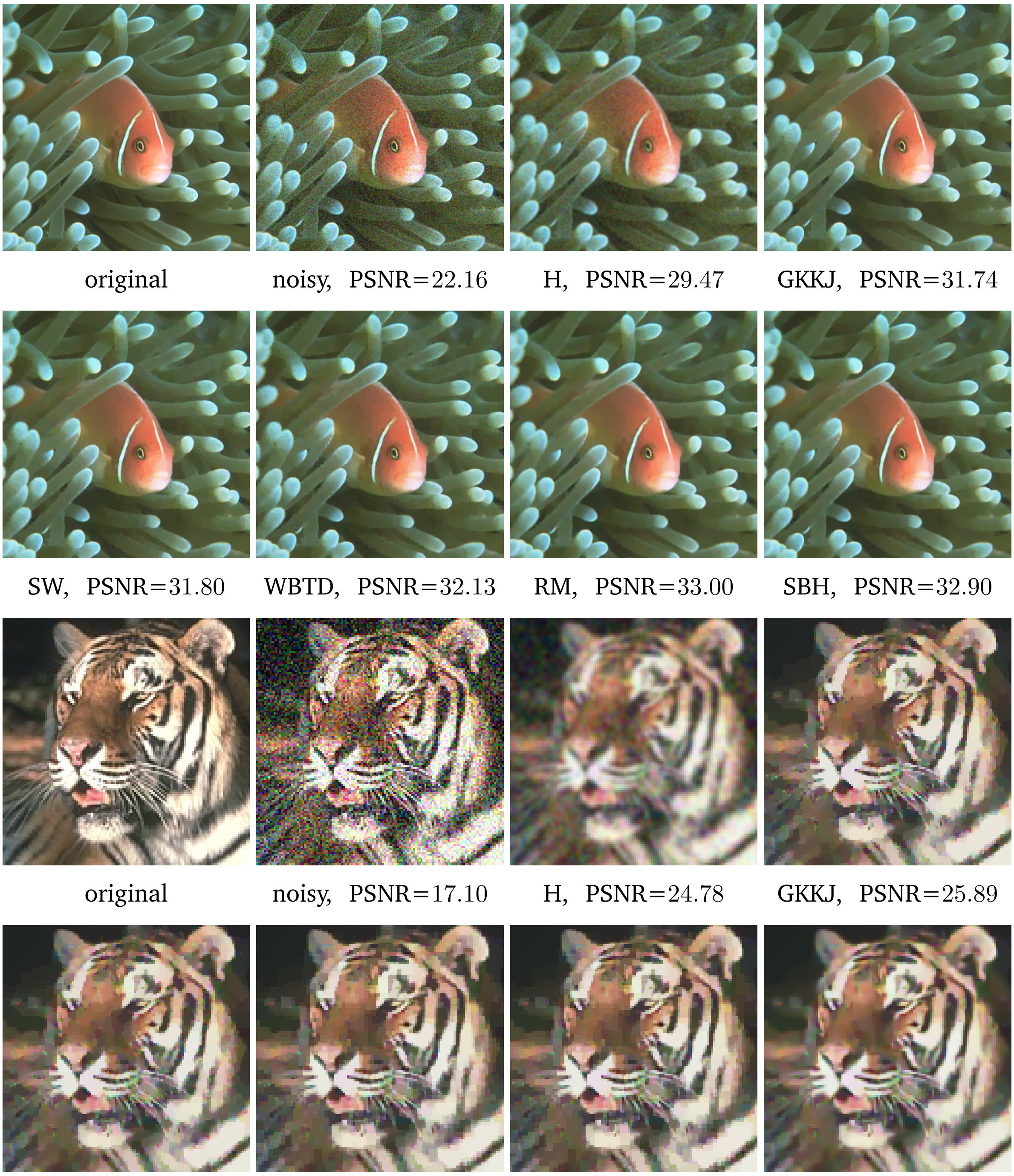


Figure 4.6.: **Estimated Diffusivities for the Berkeley Database.** The graph shows how the diffusivities corresponding to the eigenvalues μ_1 and μ_2 change with the smoothing scale ρ. For larger ρ, the diffusivities become increasingly similar and steeper on the Berkeley database for different smoothing scales.

Table 4.2.: **Denoising Results on the Berkeley Database.** Diffusivity parameters and denoising results for different diffusion models on the Berkeley test set. See Section 4.6 for the abbreviations. In the last three columns, the average PSNR for Gaussian noise with standard deviation n is given.

Model	γ_1	λ_1	γ_2	λ_2	$n = 20$	$n = 40$	$n = 60$
H	-	-	-	-	27.26	24.40	22.67
GKKJ	0.756	0.340	-	-	29.08	25.59	23.45
SW	0.754	0.336	-	-	29.09	25.64	23.52
WBTD	0.752	0.334	0.694	0.326	29.58	25.88	23.63
RM	0.733	0.304	0.644	0.231	29.56	25.99	23.76
SBH	0.739	0.315	0.646	0.234	29.66	26.05	23.80

Let us now interpret these findings from a probabilistic modelling perspective. The performance ranking according to the PSNR mirrors the accuracy of the underlying natural image priors. In particular, the large gap between homogeneous diffusion and the rest of the models is caused by the wrongly assumed Gaussian-like distribution of the underlying image prior $\mu_1 + \mu_2 = |\boldsymbol{\nabla} u|^2$ in model H (see Tab. 4.1). Since all of the remaining filters accurately reproduce the kurtotic shape of the prior distributions, they perform much better. Finally, the inherent directional bias in natural image models is only respected by the anisotropic models WBTD, RM and SBH, which gives them a consistent advantage over the isotropic models GKKJ and SW.



original — noisy, PSNR=22.16 — H, PSNR=29.47 — GKKJ, PSNR=31.74

SW, PSNR=31.80 — WBTD, PSNR=32.13 — RM, PSNR=33.00 — SBH, PSNR=32.90

original — noisy, PSNR=17.10 — H, PSNR=24.78 — GKKJ, PSNR=25.89

SW, PSNR=26.00 — WBTD, PSNR=26.21 — RM, PSNR=26.48 — SBH, PSNR=26.50

Figure 4.7.: **Denoising Results on the Berkeley Database.** Here we show two examples from the experiments from Tab. 4.2 with the algorithms L, GKKJ, SW, WBTD, RM and SBH. Visually, only the linear diffusion provides significantly worse results, the other algorithms perform on a similar level. The difference becomes more prominent for increasing standard deviation of the Gaussian noise. The PSNR is given for the whole image, but only a zoom is shown. **(a)** The first and second row show image *210088* with standard deviation $n = 20$. **(b)** The third and fourth row show image *108082* with standard deviation $n = 40$.

4.7 Natural Image Priors for Image Inpainting

> *I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail.*
>
> — **Abraham Harold Maslow**
> (Psychologist)

In the previous sections, we have only dealt with denoising. Intuitively, a natural image prior can be used for inpainting by keeping the known data constant and imposing the prior constraint on the unknown data. In this way, we could apply the *Bayesian* learning, where we fit a function to the prior distribution, directly in the compression context.

This intuitive idea can be also justified formally: For the special case of TV-inpainting, Chan and Shen [52] have proposed a variant of the Bayesian framework. Moreover, Roth and Black [237] have modified their FoE framework for inpainting. While they differ in detail, both approaches rely on the general principle of replacing the noise prior $p(\boldsymbol{f}|\boldsymbol{u})$ by a known data prior. Let $\Gamma = \{1, \ldots, n_x\} \times \{1, \ldots, n_y\}$ denote the discrete image domain. In a general formulation, such a prior models the confidence in each known point from the discrete *inpainting mask* $\tilde{K} \subset \Gamma$ in terms of a function γ:

$$p(\boldsymbol{f}|\boldsymbol{u}) = \prod_{(i,j)\in\tilde{K}} \prod_{k=1}^{n_c} \gamma(u_{k,i,j} - f_{k,i,j}). \tag{4.46}$$

Here we assume that the known data $u_{k,i,j}$ is noisy and γ represents the distribution of the noise $u_{k,i,j} - f_{k,i,j}$, where $f_{k,i,j}$ is the underlying ground truth. However, if we assume that the known data is perfect ($u_{k,i,j} = f_{k,i,j}$), we can set $\gamma \equiv 1$ and reduce our Bayesian maximisation problem to a maximisation of the image prior $p(\boldsymbol{u})$ on the missing image parts $\Gamma \setminus \tilde{K}$:

$$E_P(\boldsymbol{u}) = p(\boldsymbol{u})\Big|_{\Gamma\setminus\tilde{K}} = \prod_{(i,j)\in\Gamma\setminus\tilde{K}} \phi(\mu_{1,i,j}, \mu_{2,i,j}). \tag{4.47}$$

As in Section 4.5, we can now derive a continuous energy

$$E(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left(\frac{1}{\tau}|\boldsymbol{u} - \boldsymbol{f}|^2 + \psi(\mu_1, \mu_2)\right)\, dx\, dy \tag{4.48}$$

and the corresponding inpainting PDE

$$\partial_t u_k \;=\; \boldsymbol{\nabla}_\sigma^\top\Big((K_\rho * \boldsymbol{D})\ \boldsymbol{\nabla}_\sigma u_k\Big) \text{ on } \Omega \setminus K, \quad k \in \{1, \ldots, n_c\} \tag{4.49}$$

with Dirichlet boundary conditions on K. Here, the diffusion tensor depends on the learned prior distribution ψ as in the previous sections. This means that all models with their learned diffusivities are applicable. However, there are also alternatives for learning parameter-free diffusion models. Scharr et al. [246] use a *discriminative* approach: They perform denoising on

Table 4.3.: **Learning parameters for inpainting.** The table shows the MSE for different parameter selections in inpainting experiments with the model of Roussos and Maragos [239]. For images from the Berkeley database, 5% uniformly random distributed pixels are known for each image. We compare Bayesian and discriminative learning to ideal reference parameters for the test set of the Berkeley database. The latter were obtained with a grid search on the respective data sets.

parameter selection	γ_1	λ_1	γ_2	λ_2	*train*	*test*
Bayesian	0.7	0.3	0.6	0.2	480.92	500.7951
discriminative	0.4	0.5	0.2	0.4	**431.28**	450.01
reference	0.5	0.6	0.2	0.4	431.37	**449.96**

training data with artificial noise. Then they choose the parameters for a family of diffusion functions similar to the one from Eq. 4.45 that minimise the error w.r.t. the ground truth. Such discriminative learning is a general approach that does not rely on a general prior, but includes task-specific information.

There are also other task-specific publications that use image statistics for inpainting, but they are rare and highly specific. Levin et al. [165] fill in small missing image parts by learning the distribution of the known image content. While this approach has its merits for image restoration, it requires large amounts of image content to be known and is therefore ill-suited for our sparse compression context. Exemplar-based methods that restore missing image content according to similarities of their neighbourhood to the neighbourhood of known data can also be augmented with statistics. He and Sun [121] propose a statistical assessment of the self-similarity in images in terms of the offsets of similar patches. However, both of these methods go more in the direction of texture synthesis than the general inpainting case that we discuss in this section.

In the following, we want to compare Bayesian and discriminative learning approaches for image inpainting. In particular, we want to find out if these learning approaches could be useful for compression. To this end, we generate uniformly random distributed inpainting masks with 5% of known data for each image of the Berkeley database. Then, we consider inpainting with the anisotropic model of Roussos and Maragos [239] for image reconstruction.

As for the inpainting experiments, we use the partitioning of the Berkeley database into 100 training and 100 test images. In Section 4.6, we have already used the Bayesian approach to acquire the parameters γ_1, λ_1, γ_2, and λ_2 of our diffusivities from the training data. Now we perform simple discriminative learning in addition: We obtain our parameters by a grid search on the Berkeley training set that minimises the mean squared error (MSE) w.r.t. the reconstruction. We apply the Bayesian and discriminative parameter sets to both parts of the Berkeley database (training and test) separately. In order to evaluate how close both learned

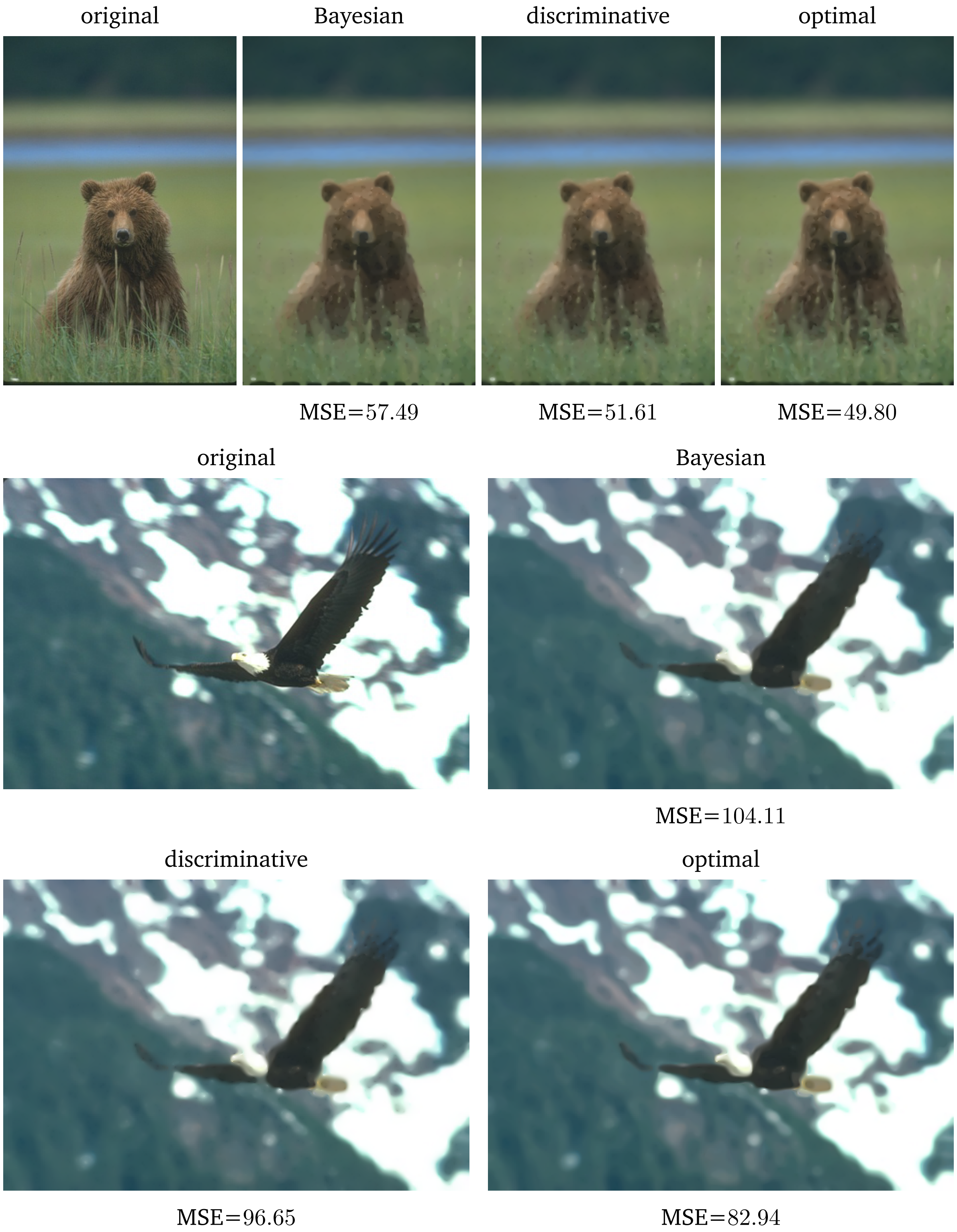


Figure 4.8.: **Inpainting with Image Specific Parameters.** Choosing optimal parameters for the images *100080* and *135037* of the Berkeley database can decrease the error by up 20% compared to learned global parameters.

models are to the optimum, we also perform a grid search for all parameters on the Berkeley test set to obtain a reference set of parameters.

Our results in Tab. 4.3 reveal that discriminative learning comes very close to the reference, while the Bayesian approach lags behind. We can identify multiple reasons for this discrepancy:

1. **Metric Awareness:** The discriminative approach minimises our error metric of choice, in this case the MSE. The natural image prior from the previous section does not incorporate this information.

2. **Mask Awareness:** Our Bayesian approach treats masks of different densities and distributions completely equally. However, it has been shown that the choice of known data influences inpainting results significantly [25, 59, 124, 184]. The discriminative model however incorporates this dependency since it relies on the distribution of known data on the training set.

3. **Global Influence:** On sparse, scattered data, the interactions of anisotropic inpainting can span large distances in the inpainting domain. Two points that are far away from each other might induce an edge somewhere inbetween. The statistics that we analyse in the Bayesian approach, however, consider mostly localised image structure. Only the smoothing scale of the gradient incorporates some neighbour information by averaging. The discriminative model however captures also these interactions.

Overall, discriminative learning produces better results than its Bayesian counterpart since it is task-specific. This comes at the cost of an increased run-time: Our simple grid-search approach is considerably slower than Bayesian learning (several days on a cluster compared to minutes on a single machine). This drawback is however not significant: On one hand, more sophisticated optimisation algorithms might speed up the discriminative approach significantly in the future. On the other hand, speed is not that relevant for offline learning, since it only has to be performed once.

If we want to apply learning for compression, we have to consider one important difference to the inpainting situation that we have ignored so far: In compression, we have direct access to the ground truth. This allows us to choose both the known data and the diffusion model freely. By definition, parameters adapted to the ground truth can never be worse for the reconstruction of the original than parameters optimised for a training data set. In Figure 4.8 we consider inpainting on individual images of the Berkeley database with the model of Roussos and Maragos [239]. This time, we perform a grid search to optimise the parameters for two individual images of the database and compare the results to the globally optimised parameters from above. In the case of image *100080* of the Berkeley database (bear), optimal parameters are $\gamma_1 = \gamma_2 = 0.4$, $\lambda_1 = 0.1$, $\lambda_2 = 0.9$. For image *135037*, we get $\gamma_1 = 0.6$, $\gamma_2 = 0.4$, $\lambda_1 = 0.1$, $\lambda_2 = 0.3$ instead.

As expected, the individual parameter choices outperform both the Bayesian and the discriminative training by up to 20%. Overall, our experiments lead to the following conclusion: Discriminative learning can speed up compression methods by removing the need for parameter optimisation at the cost of reduced reconstruction quality. However, if the goal is optimal accuracy, an optimisation that considers the original image is still necessary.

4.8 Towards a Statistical Interpretation of EED

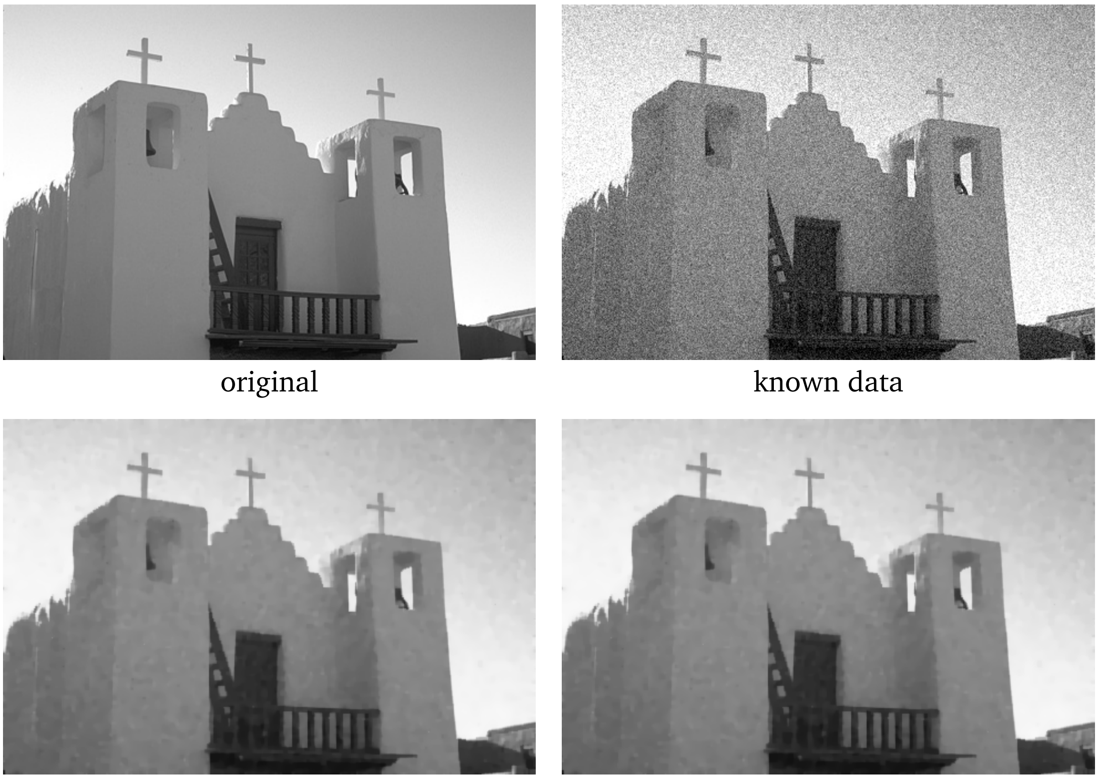
> *It is better to have an approximate answer to the right question than an exact answer to the wrong one.*
>
> — **John Wilder Tukey**
> (Mathematician)

In the previous sections, we have shown that anisotropic diffusion models capture the distribution of natural priors better than isotropic ones. Therefore, they should also be more suitable for compression applications. Experiments by Schmaltz et al. [252], where multiple inpainting operators are evaluated, confirm this assumption. However, successful image compression methods like the R-EED codec from Chapter 3.3 use EED, a diffusion model that is not covered by our unifying statistical prior framework. For compression, EED seems particularly attractive since it is powerful and, at the same time, has a simple structure: The eigenvalue that corresponds to diffusion along the edge is fixed to one. For the direction across the edge, a Charbonnier diffusivity with a single contrast parameter λ is used in compression. In contrast, our anisotropic models from the previous section have four free parameters.

It is difficult to include EED in a common statistical framework, since so far no corresponding energy functional has been found. Tschumperlé [282] has even argued that no EED-functional that is based on the eigenvalues of the structure tensor exists. However, this does not disprove the existence of another variational formulation of EED. Krajsek and Scharr [158] have tried to connect EED to priors by employing an intermediate step that computes the diffusion tensor on the result of an independent process. In the following we provide an incentive for future work by showing a relationship between the EED model and the statistics of directional derivatives.

To this end, we do not use the structure tensor to derive natural image priors as in the previous sections. Instead, we consider the directional derivatives across edges (in the direction of the smoothed gradient $\boldsymbol{\nabla} u_\sigma$), and along image edges (orthogonal to $\boldsymbol{\nabla} u_\sigma$):

$$\partial_{\boldsymbol{\nabla} u_\sigma} u := \frac{\boldsymbol{\nabla} u_\sigma^\top}{|\boldsymbol{\nabla} u_\sigma|} \boldsymbol{\nabla} u, \qquad \partial_{\boldsymbol{\nabla} u_\sigma^\perp} u := \frac{\boldsymbol{\nabla} u_\sigma^{\perp\top}}{|\boldsymbol{\nabla} u_\sigma|} \boldsymbol{\nabla} u. \tag{4.50}$$



original

known data

EED, MSE=31.67

DDD, MSE=31.45

Figure 4.9.: **Denoising with Directional Derivative Diffusion.** Both EED and DDD use a small smoothing scale $\sigma = 0.1$, a Charbonnier diffusivity and a contrast parameter $\lambda = 0.5$. Note that EED sets the diffusivity along edges to 1, while DDD also uses a Charbonnier diffusivity.

In the following, we propose a new directional derivative diffusion (DDD) model. As before, we construct a continuous energy of the form

$$E(u) = \int_{\Omega} \left(\psi_1\Big((\partial_{\boldsymbol{\nabla} u_\sigma} u)^2 \Big) + \psi_2\big((\partial_{\boldsymbol{\nabla} u_\sigma^\perp} u)^2 \big) \right) dx\,dy. \tag{4.51}$$

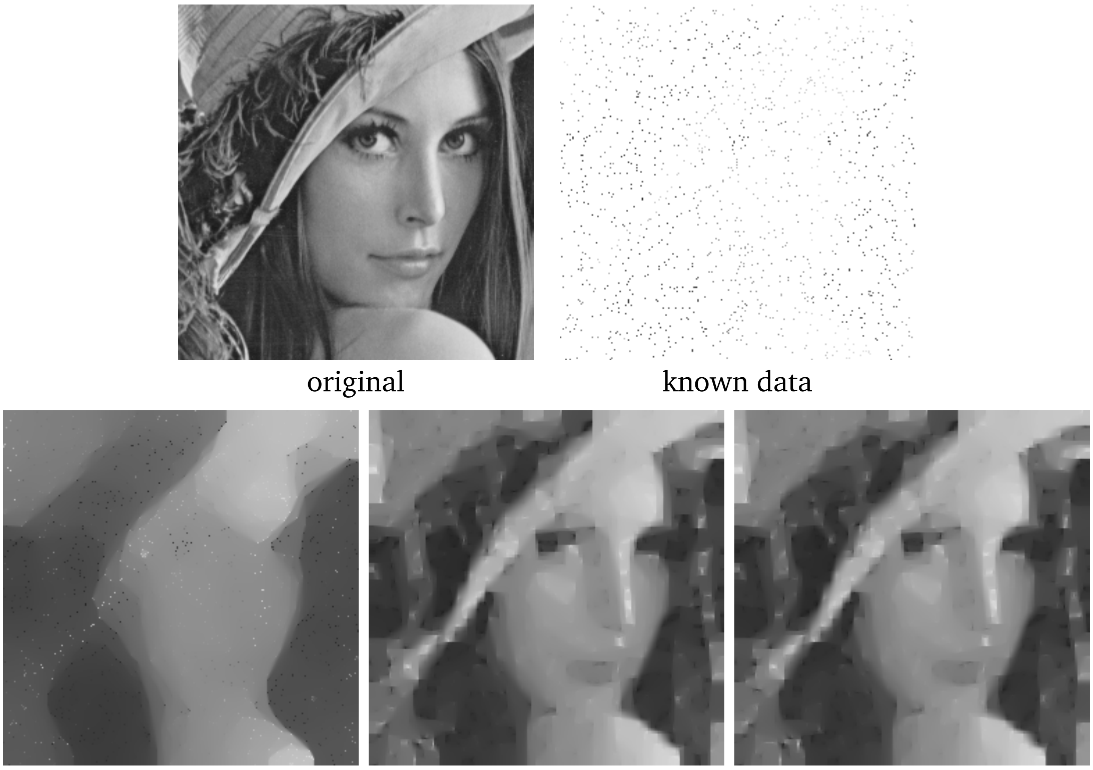
In Appendix G.2 we show that the Euler-Lagrange equations of this functional lead to the diffusion process

$$\partial_t u = \operatorname{div}\left(\boldsymbol{D}\boldsymbol{\nabla} u\right) + \operatorname{div}\left(K_\sigma * (\hat{\boldsymbol{D}}\boldsymbol{\nabla} u) \right) \tag{4.52}$$

with the EED-like diffusion tensor

$$\boldsymbol{D} := \psi_1'\left((\partial_{\boldsymbol{\nabla} u_\sigma} u)^2 \right) \frac{\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\top}{|\boldsymbol{\nabla} u_\sigma|^2} + \psi_2'\left((\partial_{\boldsymbol{\nabla} u_\sigma^\perp} u)^2 \right) \frac{\boldsymbol{\nabla} u_\sigma^\perp \boldsymbol{\nabla} u_\sigma^{\perp\,\top}}{|\boldsymbol{\nabla} u_\sigma|^2}. \tag{4.53}$$

and another anisotropic tensor $\hat{\boldsymbol{D}}$ that potentially can have negative eigenvalues. The second term can therefore lead to backward diffusion that creates difficult numerical challenges. Nevertheless, the model features some striking similarities with the original EED model that we have discussed in Chapter 3.1.



original

known data

isotropic, $\mathrm{MSE}=1162.07$

EED, $\mathrm{MSE}=586.74$

DDD, $\mathrm{MSE}=594.73$

Figure 4.10.: **Inpainting with Directional Derivative Diffusion.** Both EED and DDD use a small smoothing scale $\sigma = 0.1$, a Charbonnier diffusivity and a contrast parameter $\lambda = 0.1$.

For $\sigma = 0$, we show in Appendix G.2 that the equation can be significantly simplified and one obtains the isotropic case

$$0 = -\mathrm{div}\left(\psi_1'(|\boldsymbol{\nabla} u|^2)\boldsymbol{\nabla} u\right), \tag{4.54}$$

which is consistent with EED and the other anisotropic models from the previous sections.

While a thorough analysis of this model is beyond the scope of this thesis and its focus on compression, we provide some experimental results that indicate potential for future work. For denoising in Figure 4.9 and inpainting in Figure 4.10 with small values of σ, DDD and EED produce virtually the same results. In particular, the inpainting experiment in Figure 4.10 shows that this similar behaviour does not merely stem from the fact that DDD and EED both approach the same isotropic model for small smoothing scales. In contrast to isotropic diffusion, both models are still able to produce sharp edges in the inpainting domain.

4.9 Conclusion

> *I have had my results for a long time,*
> *but I do not yet know how to arrive at them.*
>
> — **Carl Friedrich Gauß**
> (Mathematician and Physicist)

This chapter contributes to this thesis in two different regards: On one hand, we have presented a new unifying statistical framework for diffusion models that allows general theoretical insights regarding the systematic relation of probabilistic priors, variational methods, and PDEs. On the other hand, we have justified the use of anisotropic diffusion in compression.

Our unifying framework contains eight diffusion filters that have a corresponding variational formulation. It enabled us to derive these models from probabilistic filters with a structure tensor prior. We have verified experimentally that those filters which model the structure adaptive statistics of natural images more accurately also offer a better performance in practice. This justifies their use in digital image processing and computer vision, and it establishes a hitherto unknown reason for the success of anisotropic filters. In particular, we can now interpret the evolution of diffusion filters according to a problem-driven design as an increasingly accurate representation of the structure tensor's statistics. From a statistical point of view, we have emphasised the importance of directional statistics that take into account the local image structure and its scale dependency. Interestingly, our statistical foundation of tensor-driven diffusion gives also additional insights that go beyond a pure statistical foundation of existing models: For instance, it sheds light on how the decay function of each eigenvalue should be adapted to the smoothing scales of the structure tensor.

In regards to the specific task of image compression, the statistical analysis provides another justification of anisotropic diffusion in image compression. While we have not found an exact relationship between the EED model from the R-EED codec and a Bayesian prior, it is closely related to locally adaptive features such as the structure tensor and directional derivatives. Our Bayesian approach yields valuable theoretical insights, but our experiments have shown that its use for parameter learning is limited in compression applications. Since we have access to the ground truth images in compression, we still optimise the diffusion parameters w.r.t. an optimal reconstruction in the following chapters.

Publication Info

The unifying diffusion framework was presented by Peter, Weickert, Munk, Krivobokova, and Li [224] at the Energy Minimisation Methods in Computer Vision and Pattern Recognition Conference 2015.

5 Quality: Understanding PDE-based Compression

> *The purpose of computing is insight, not numbers.*
>
> — **Richard W. Hamming**
> (Mathematician)

For a given image, the core task of lossy image compression is to store a small amount of data from which the original can be reconstructed with high accuracy. In Chapter 4, we have concluded that inpainting with nonlinear anisotropic PDEs such as edge-enhancing anisotropic diffusion (EED) [295] is particularly well-suited for this task due to its accurate approximation of the statistics of natural images. The R-EED codec by Schmaltz et al. [252] marks the current state of the art in PDE-based compression. Its reliance on EED is in accordance with our previous assessment of inpainting operators.

The drawbacks of nonlinear anisotropic inpainting in comparison to linear isotropic models such as homogeneous or biharmonic PDEs are a higher computational cost and the requirement for parameter optimisation. Simple homogeneous and biharmonic inpainting can also restore images with a high quality from a small fraction of prescribed image points, if position and value of these known data are carefully optimised [25, 59, 124, 184]. This suggests the viability of linear, parameter-free PDEs for image compression. Unfortunately, all these data optimisation efforts are in vain, as long as one cannot store the data efficiently. The optimised known data is expensive to store. Therefore, homogeneous diffusion has only been successful in specialised applications which do not employ the sophisticated data optimisation methods mentioned above (e.g. cartoon compression by Mainberger and Weickert [181]). So far, there is no full compression codec based on biharmonic inpainting at all. This raises the following question: Are competitive codecs with linear inpainting PDEs possible for general image content or is anisotropic diffusion the only viable option?

Our review of prior diffusion-based compression methods in Chapter 3.2 has already revealed that the success of a codec does not rely purely on a good inpainting technique. We have identified four key elements of PDE-based compression that every codec has to address: *inpainting*, *selection of known data*, *representation of spatial locations*, and *efficient storage*. In the following, we investigate the interplay between these components and assess linear isotropic and nonlinear anisotropic inpainting techniques in the specific context of compression.

Our contributions. In order to evaluate harmonic, biharmonic and EED inpainting, we propose two new codecs:

1. A compression codec that combines *free choice of known image points* with tailor-made entropy coding. For linear diffusion, we use an optimal control scheme [59, 124] to find the position of known data. For EED, we rely on the stochastic method of Mainberger et al. [184]. For the first time, we embed these powerful data optimisation strategies into a complete compression codec.

2. A new stochastic method that restricts pixel selection to a *locally adaptive grid*. This allows to store these positions efficiently as a binary tree at the expense of reconstruction quality.

We evaluate how individual restrictions and lossy compression steps of both frameworks affect the performance of harmonic, biharmonic, and EED inpainting. In addition, we compare our best methods against the state of the art in PDE-based compression and the quasi-standards in transform-based compression.

Related work. In Chapter 3.2.3, we have already provided a detailed overview over diffusion-based compression algorithms and related inpainting techniques. Thus, in the following, we only discuss the contributions that are specifically relevant for this chapter.

Homogeneous diffusion has been applied for the compression of specific classes of images. In particular, Mainberger et al. [183] have proposed a highly efficient codec for cartoon images, and there are several successful coders for depth maps [103, 127, 166]. However, unlike our approach, these methods rely primarily on semantic image features such as edges. This choice is motivated by the theoretical results of Belhachmi et al. [25], which suggest to choose known data at locations with large Laplacian magnitude. Köstler et al. [156] apply homogeneous diffusion for real-time video playback on a Playstation 3.

Well-performing codecs with PDEs for general image content are mainly based on EED [295] and efficient representations of data locations by binary trees. Initially, this class of methods was proposed by Galić et al. [100], while the current state of the art is the R-EED codec by Schmaltz et al. [252]. Modifications and extensions of R-EED include colour codecs [219], 3-D data compression [252], and progressive modes [251].

In addition, there are several works that are closely related to compression, but do not consider actual encoding [25, 59, 124, 125, 184]. Instead, they deal with optimal reconstruction from small fractions of given data. We directly use results from the optimal control scheme for harmonic PDEs by Hoeltgen et al. [124] and its biharmonic extension. Our densification approach on restricted point sets is inspired by the approach of Mainberger et al. [125, 184]. They consider a stochastic sparsification on unrestricted point sets which has the advantage that it can be applied for any inpainting method without the need for modifications. For a more detailed discussion of PDE-based methods we refer to Chapter 3.2.

Organisation of the chapter. We begin with a review of different optimisation techniques for image inpainting in Section 5.1. This covers both the selection and representation of locations for known pixels, as well as the optimisation of pixel values. In Section 5.2 we review different approaches for efficient storage of known data that act as the foundation for our new codecs. These approaches include entropy coding as well as additional preprocessing steps such as prediction.

We introduce our two new compression codecs in Section 5.3. They are specifically designed to allow a practically relevant evaluation of different inpainting PDEs in the compression context. Based on these codecs, we perform a detailed experimental analysis in Section 5.4. This allows us to evaluate both the overall performance of PDEs in compression, as well as the influence and interaction of individual compression steps. Section 5.5 concludes this chapter with a summary and outlook on future work.

5.1 Optimising and Representing Known Data

> *Location, location, location.*
>
> — **Lord Harold Samuel**
> **(according to etymologists)**
> (Real Estate Tycoon)

Many research results have confirmed that the choice of known data influences the reconstruction significantly [25, 59, 124, 184]. At the same mask density, choosing optimal positions and pixel values is vital for a good reconstruction. In this thesis, we consider a separate optimisation process for both types of data. First we optimise the locations of data and then choose the corresponding pixel values in a so-called tonal optimisation step. In this section we provide a review of optimisation methods that are relevant for our new codecs in Section 5.3.

Image Reconstruction. First we briefly recapitulate the PDE-based reconstruction setting in its general formulation from Chapter 3.1: The greyscale image $f : \Omega \to \mathbb{R}$ is known on the inpainting mask $K \subset \Omega$, and we want to reconstruct the missing data in the inpainting domain $\Omega \setminus K$. For a suitable differential operator L, we obtain the missing image parts u as the steady state for $t \to \infty$ of the evolution that is described by the PDE

$$\partial_t u = Lu \quad \text{on } \Omega \setminus K. \tag{5.1}$$

Here, we impose reflecting boundary conditions at the image boundary $\partial\Omega$. In addition, the known data is fixed on K, thus creating Dirichlet boundary conditions $u = f$. In the following, we consider two different parameter-free choices for the differential operator L. In the simplest case, we apply *homogeneous diffusion* [141]:

$$Lu = \Delta u = \text{div}(\boldsymbol{\nabla} u). \tag{5.2}$$

Since experiments suggest that the *biharmonic operator*

$$Lu = -\Delta^2 u \tag{5.3}$$

may give better reconstructions [59, 100, 252], it is also considered. Both operators propagate known information equally in all directions throughout the whole image evolution. Finally, we also use EED with

$$Lu = \text{div}(\boldsymbol{D}\boldsymbol{\nabla} u) \tag{5.4}$$

and an anisotropic tensor $\boldsymbol{D}$ that adapts to the local image structure as described in Chapter 3.1. In principle, this allows EED to obtain more accurate reconstructions from the same amount of known data [100, 252]. However, the price for this increase in quality are algorithms with higher computational complexity and the need for parameter optimisation. Therefore, successful compression codecs like R-EED [252] rely on heuristics to optimise known data (see Chapter 3.3).

For the purpose of the optimal control approaches that we will discuss in the following, it is also helpful to recall the alternative, more general formulation of the inpainting problem that we have discussed in Chapter 3.1. The general inpainting equation (3.25)

$$(1 - c(\boldsymbol{x}))Lu - c(\boldsymbol{x})(u - f) = 0 \tag{5.5}$$

uses a confidence function $c(\boldsymbol{x}) : \Omega \to [0, 1]$ to balance closeness to the original data and the smoothness constraint imposed by the operator L. For a binary confidence function, this model is equivalent to the formulation in Eq. 5.1, as elaborated in more detail in Chapter 3.1.

5.1.1 Spatial Optimisation

Finding optimal positions for a fixed amount of mask points is nontrivial. Let us consider a simple example: we have a discrete image with a resolution of 256×256 and want a sparse representation that contains only 5% of the total number of pixels. Combinatorics tells us that there are

$$\begin{pmatrix} 256^2 \\ 0.05 \cdot 256^2 \end{pmatrix} = \begin{pmatrix} 65536 \\ 3276.8 \end{pmatrix} \approx 9.5 \cdot 10^{5647} \tag{5.6}$$

possibilities to select this amount of known data. This large search space increases both with the resolution of the original image and the percentage of selected pixels. In the following, we review the three most successful approaches to solve this problem in image compression.

Optimal Control. Hoeltgen et al. [124] and Chen et al. [59] optimise the locations of the inpainting mask by solving a constrained optimisation problem of the form

$$\underset{u,\,c}{\operatorname{argmin}}\, E(u,c) = \frac{1}{2}\int_{\Omega}\Big((u(\boldsymbol{x})-f(\boldsymbol{x}))^2 + \delta|c(\boldsymbol{x})| + \varepsilon c(\boldsymbol{x})^2\Big)\,d\boldsymbol{x}, \tag{5.7}$$

$$\text{such that}\quad c(\boldsymbol{x})(u(\boldsymbol{x})-f(\boldsymbol{x})) - (1-c(\boldsymbol{x}))Lu(\boldsymbol{x}) = 0. \tag{5.8}$$

Note that at the image boundaries, reflecting boundary conditions still apply. Here the goal is to find simultaneously the reconstruction u and the real-valued *confidence function* $c(\boldsymbol{x})$ by minimising the energy $E(u,c)$. The general inpainting equation (3.25) acts as a side-constraint. The term $(u(\boldsymbol{x})-f(\boldsymbol{x}))^2$ penalises deviations of the reconstruction u from the original f, while term $|c(\boldsymbol{x})|$ imposes sparsity of the confidence function. Thereby, the parameter $\delta > 0$ can be used to determine the amount of known data that influences the reconstruction. The other parameter, $\varepsilon > 0$, is fixed to a small positive value, since the existence of a solution for efficient solvers is not guaranteed for $\varepsilon = 0$ (see Hoeltgen et al. [124]).

In fact, minimising the energy $E(u,c)$ is challenging due to two facts: the sparsity term is nondifferentiable and the problem is nonconvex. Nevertheless, efficient algorithms are possible by considering a series of related linear problems from optimal control theory. It has been shown that fast primal-dual schemes can be employed for finding a solution [59, 124]. While Hoeltgen et al. [124] focus on homogeneous diffusion, biharmonic inpainting is considered in addition by Chen et al. [59].

Note that for compression, a continuous confidence function is disadvantageous, since the coding cost of real numbers is much higher than for the integer case. However, Hoeltgen and Weickert [126] have shown that there is no drawback, if the continuous confidence function is reduced to a binary mask by thresholding. The same reconstruction quality as with a continuous confidence function can be achieved, if one optimises not only the location, but also the value of the known pixels. We discuss such *tonal optimisation strategies* at the end of this section.

Stochastic Sparsification. While optimal control approaches are mathematically well-founded and can be implemented efficiently, they also have two drawbacks: On the one hand, there is no straightforward extension to nonlinear anisotropic diffusion, so far. On the other hand, the parameter δ controls the amount of known data only indirectly. It has to be tuned to achieve a specific density of the inpainting mask.

Instead, one can employ the *stochastic sparsification* approach of Mainberger et al. [184]. It starts with a full mask that contains all image points. From this mask, we remove a fixed percentage α of known data. After inpainting with the smaller mask, we add a fraction β of

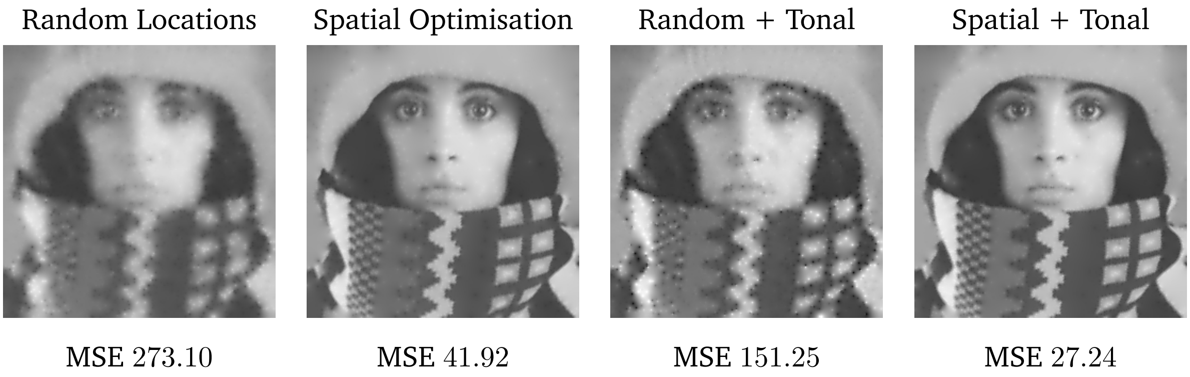


Figure 5.1.: Spatial and tonal optimisation example with homogeneous diffusion on the test image *trui*. In all cases, 4% of the image pixels are known. Optimising the locations of known data improves the quality of homogeneous diffusion more than keeping random locations and applying tonal optimisation. For the best results, a tonal optimisation is applied after selecting optimal locations with a nonlocal pixel exchange. Images courtesy of Hoeltgen et al. [125].

the removed pixels with the highest reconstruction error back to the mask. This sparsification algorithm iterates the aforementioned steps until the target mask density is reached.

However, there is a substantial risk that this algorithm is caught in a local minimum. To avoid this problem, Mainberger et al. [184] propose a *nonlocal pixel exchange* for postprocessing: First, they remove n randomly selected points from the mask and reconstruct the image. Afterwards, a candidate set of $m > n$ non-mask pixels is selected randomly. From this candidate set, the algorithm adds the n points that have the largest reconstruction error back to the mask. If the new mask yields a better reconstruction it is kept, otherwise the change is reverted.

Optimal control and stochastic sparsification yield results of comparable quality, but stochastic sparsification is usually significantly slower. An introductory example for the effect of spatial optimisation can be found in Figure 5.1.

Tree-based Subdivision. In the context of the R-EED codec by Schmaltz et al. [252], we have described another approach to spatial optimisation in Chapter 3.3: A *subdivision* algorithm partitions the original image into rectangular subimages. For each subimage, it selects the corner points and its midpoint as known data. If the local reconstruction error in a subimage exceeds a user-defined error threshold, the image is subdivided, thus adding more known data in regions that are not reconstructed well. The advantages of this method are twofold: On one hand, it reduces the coding cost of the locations since the subdivision can be efficiently represented by a binary tree. On the other hand, it also reduces the size of the search space.

The obvious drawback of *tree-based approaches* is that the restriction to a locally adaptive grid might also reduce the reconstruction quality. In this chapter, we also want to investigate how such restrictions affect different inpainting PDEs. We cannot rely on a heuristic subdivision strategy, since this might skew the comparison to *exact masks* that arise from optimal control

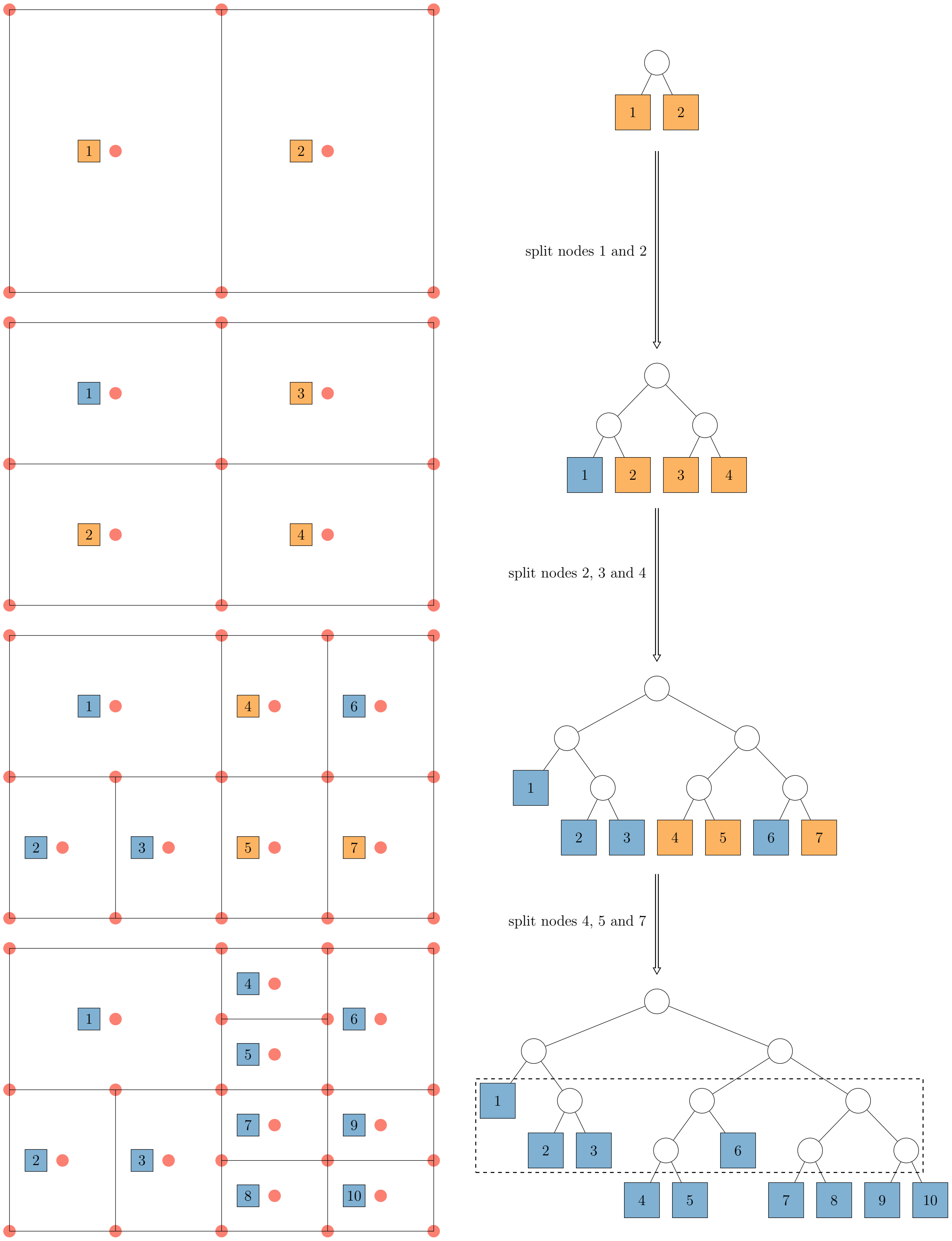


Figure 5.2.: **Rectangular Subdivision Scheme.** Each square-shaped leaf node in a tree corresponds to a subimage with the same number. Red circles represent the locations stored by each subimage. Nodes with orange colour exceed the local error threshold and must be split in the next step. The final tree has minimum depth 2 and maximum depth 4. Therefore, only the levels 3 and 4 (marked by dashed box) need to be stored: each leaf node as a 0, each inner node as a 1. This yields the overall binary representation 0111001011 with 4 bits for level 3 and 6 bits for level 4.

or stochastic sparsification. Therefore, we propose a new tree-based stochastic approach in Section 5.3. We discuss the tree representation that we need for this purpose in more detail in the following paragraphs.

For a tree T consisting of nodes $t_0, \dots, t_n$, the root node t_0 stands for cutting the original image in half in its largest dimension. By adding nodes to the tree, one of the two subimages corresponding to the parent node is split further. In order to encode the tree efficiently in a binary sequence, we exploit that *leaf nodes* are indicators for termination, i.e. the subimage corresponding to a leaf node is not split any further. Thereby, we can represent the tree as a bit sequence (0 for leaf nodes, 1 for other nodes) that results from traversing the tree level by level. We can reduce the coding cost even further by storing the *minimum and maximum tree depth*. All nodes on tree levels up to the minimum depth are split, and all nodes on levels above the maximum tree depth are leaf nodes. Thus, only the tree structure for the levels in-between these depths needs to be encoded. Figure 5.2 provides a visualisation of the subdivision scheme and tree representation. In particular, it also gives a concrete example for a conversion of a tree to a binary sequence.

5.1.2 Tonal Optimisation

Tonal optimisation is the task of choosing optimal pixel values (i.e. grey or colour values) for an inpainting mask with fixed locations. This optimisation in the co-domain complements the spatial optimisation in the image domain from the previous section. Intuitively, tonal optimisation can be understood as introducing a small error to the sparse known data to achieve a more significant improvement in the inpainting domain.

In the linear case, tonal optimisation can be formulated as a least squares problem [125]. Let $\boldsymbol{f} \in \mathbb{R}^{n_x n_y}$ denote the original image in vector notation, $\boldsymbol{c} \in \{0,1\}^{n_x n_y}$ the corresponding binary mask, and $\boldsymbol{r}(\boldsymbol{c},\boldsymbol{g})$ the reconstruction that one obtains with linear diffusion inpainting from the mask $\boldsymbol{c}$ and the known data $\boldsymbol{g}$. Then, optimal known data can be found by the minimisation

$$\underset{\boldsymbol{g}}{\operatorname{argmin}}\, |\boldsymbol{f} - \boldsymbol{r}(\boldsymbol{c},\boldsymbol{g})|^2. \tag{5.9}$$

Hoeltgen et al. [125] have shown that this problem has a unique solution, if the mask is not empty and a linear PDE is used for reconstruction. There are many ways to find this solution. Originally, Mainberger et al. [184] have proposed a randomised Gauß-Seidel scheme that relies on so-called inpainting echoes. Setting a single point of the inpainting mask to 1 and all other mask points to 0 yields the influence of this mask point on the reconstruction, its *echo*. Mainberger et al. [184] have shown that the full inpainting can then be represented as a weighted sum of these echoes, which makes inpainting extremely fast, once these echoes have been computed. However, computing the individual echoes takes time and they are only valid for a specific mask configuration.

More recently, other methods have been proposed that circumvent the costly computation of the echoes. For example, Chen et al. [59] and Hoeltgen and Weickert [126] apply primal-dual methods to solve the problem directly. Hoeltgen et al. [125] propose a gradient descent algorithm that is accelerated with fast explicit diffusion (FED) [114]. In Section 5.3 we argue that for our specific use in compression, the original echo-based approach is still viable compared to these newer, more efficient methods. Moreover, note that the least squares formulation yields continuous optimal grey values that are just as costly to store as the continuous confidence function from the previous section. We address this problem also in Section 5.3.

Unfortunately, the aforementioned solvers are not directly applicable to nonlinear anisotropic diffusion. Instead we apply a straightforward iterative algorithm that is also used in R-EED [252]: We visit all mask pixels in random order and check if increasing or decreasing the pixel value by fixed steps yields an improvement. If it does, we keep the new pixel value, otherwise we revert to the original. We iterate these random walks over the whole mask. Similar to the probabilistic sparsification algorithm, this method is rather slow, but universally applicable to a wide range of inpainting operators.

5.2 Storing Data Efficiently

> *Chaos is merely order waiting to be deciphered.*
>
> — **José de Sousa Saramago**
>
> (Journalist, Author, Winner of the Nobel Prize)

In this section we discuss general requirements and restrictions that the compression setting imposes on optimisation algorithms. This enables us to design codecs that offer a good trade-off between file size and inpainting quality.

The most important ingredient for efficient storage of known data in PDE-based compression is entropy coding. In Chapter 3.2.1, we have discussed the information theoretical background of entropy coders and given a general historical overview. Here, we present specific entropy coders in more detail and also provide an overview over preprocessing steps that can enhance the efficiency of entropy coders.

5.2.1 Entropy Coding

All entropy coders share the common goal of removing redundancy from data. Thereby, they store information losslessly, but with a reduced file size. Huffman coding [136], adaptive arithmetic coding [234], and PAQ [178] have all been successfully used in PDE-based compression

[252]. So far, the primary task of these encoders has been to encode the known pixel values. Pixel locations that are represented by a binary tree have only very little potential for further lossless compression. Scheer [247] has shown that even with considerable effort, reductions of the file size are small.

Huffman Coding assigns binary codes of variable length to each grey value with the help of a binary tree. Each grey value corresponds to the leaf node of the tree and their frequency of occurrence in the image is proportional to their tree level. Following the path from the root to a leaf and adding a 1 or a 0 to a binary string depending on whether the left or the right edge is taken yields the code for the grey value associated with the leaf node. Huffman coding yields an optimal average code word length from the perspective of information theory. However, it requires at least one bit per grey value and the tree has to be transmitted as overhead. Otherwise, decoding is not possible.

Arithmetic Coding. The limitation of 1 bit per grey value can be circumvented by arithmetic coding. It assigns a single codeword to the whole sequence of grey values instead of generating a codeword for each individual grey value. It starts with an arbitrary interval $[a, b]$ with $a, b \in \mathbb{R}$. Then, we partition this interval into subintervals, one for every possible grey value. The length of these subintervals is proportional to the frequency of occurrence of the corresponding pixel value. Now, the subinterval of the first grey value that needs to be encoded is subdivided in the same way. This procedure is now repeated: Each additional grey value restricts the interval further. The final codeword is then a binary representation of an arbitrary dyadic fraction from this interval.

PAQ. Compression performance can be improved further by considering not only the frequency of occurrence of grey values, but also to exploit patterns in the sequence. PAQ is a highly evolved version of prediction by partial matching: It compresses a binary stream by predicting with a very high accuracy if the next bit is a 0 or a 1. To this end, it relies on a large number of complex context models that track how often certain patterns occur in the file. All of these context models are then mixed in a neural network that adapts to the local content of a file during encoding. Such context mixing allows to compress files with varying content very efficiently. The adaptation of context weights is performed by a gradient descent on the coding cost that is computed after a bit is encoded: At this point it becomes clear if the prediction was right or wrong.

Schmaltz et al. [252] have already conducted an evaluation of the aforementioned entropy coders. They have concluded that arithmetic coding and PAQ offer the best compression results. In our setting, PAQ appears to be particularly interesting due to its ability to adapt to the local content of a file. If we want to store pixel-accurate locations, this comes down to storing a binary image in addition to the sequence of grey values. In contrast to the tree representation in R-EED, such a binary image contains much more redundancy. Therefore, PAQ can be directly applied as an efficient container format for both positional and brightness data.

5.2.2 Storing Binary Images

We have tried many different methods that are specialised on the encoding of sparse, binary patterns. In the following we give a short overview of these methods. First, there are off-the-shelf codecs that specialise in binary images.

JBIG uses lossless compression by prediction and arithmetic coding [146]. It considers a fixed configuration of image points that optionally also includes a single pixel with variable position as a context. Depending on the frequency of occurrence of black or white pixels in this context, it predicts the colour of the next pixel. Arithmetic coding is the most efficient entropy coder supported by JBIG.

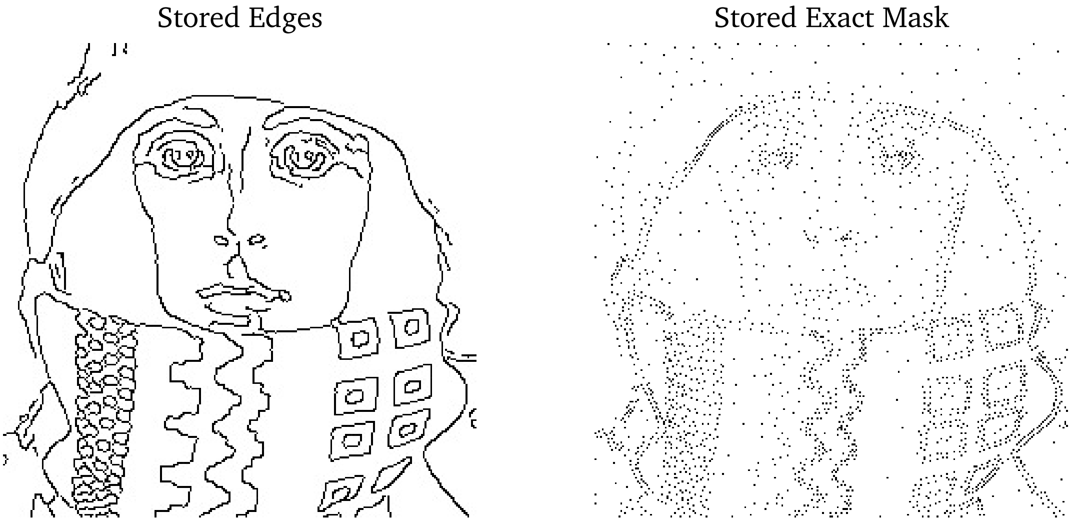
JBIG2 extends the ideas of JBIG by introducing pattern matching and dictionary approaches [131]. It is specifically tailored to text and half-toned images. To this end, it creates a dictionary of repetitive patterns that occur frequently in the image. This dictionary is then used for efficient lossless or lossy compression. For the storage of patterns that do not fit the dictionary, JBIG2 falls back to JBIG encoding.

DjVu is a collection of different compression algorithms that work in tandem [31]: For encoding of mixed content (e.g. images and text), it decomposes images in a binary image foreground and a grey- or colour-valued background part. It compresses the background with wavelet-based compression (IW44 codec) and the foreground with an encoder called JB2. JB2 is a modified version of JBIG2 that relies on the same core concepts.

In PDE-based compression, the aforementioned codecs have been already compared in the context of cartoon compression [183]. Nevertheless, we perform a new evaluation for our compression framework in Section 5.3, since the nature of our binary images is different. Mainberger et al. [183] compress edge images that feature a lot of connected lines while our optimised masks consist mainly of scattered individual pixels (see e.g. Figure 5.3). For the same reason, we also consider different approaches for storing binary images, namely block coding techniques.

ZA Block Coding by Zeng and Ahmed [321] is specifically designed for sparse binary patterns. It transforms the image into a sequence of binary values by traversing the image row by row. This sequence is then divided into blocks of length b. During encoding, it separates each block by a 0-bit and encodes the relative position of each 1 in the block. The number of bits necessary to encode these positions is given by the block length. Furthermore, each stored coordinate has a leading 1-bit such that the end of block can be detected.

MF Coordinate Coding decomposes the image into non-overlapping rectangles that contain only 1's (black pixels). These rectangles can be stored in terms of the coordinates of



Stored Edges

Stored Exact Mask

Figure 5.3.: Examples of edges stored in semantic approaches (left, courtesy of Mainberger et al. [183]) and exact masks (right) for the test image *trui*. Since the edge-image contains a lot more connected components and is well-structured, efficient encoding with JBIG is possible. In contrast, the scattered data in the exact mask is more challenging to compress.

two opposing corner points [196]. The algorithm proceeds row by row: if a row does not contain black pixels, a single 0 indicates that the row is empty and nothing must be stored in addition. A 1 initiates a non-empty row that consists of a sequence of zeroes and special symbols for the top left corner (11) and the bottom right corner (10) of a rectangle. Isolated points are encoded with the sequence 100.

ZN Pattern Coding by Zahir and Naqvi [320] combines ideas from block coding [321] and coordinate coding [196]. It reduces code-word lengths by decomposing an image into rectangles and isolated pixels. Furthermore, it employs image flipping to reduce the length of stored relative coordinates for images with a directional bias. Finally, it employs block coding in the style of [321] to store the isolated pixel part of the image.

Hierarchical Block Coding divides the image into blocks of a prescribed size [97]. It then indicates empty blocks (containing only zeroes) by a single 0. Otherwise, it stores a 1 and divides this blocks into 4 non-overlapping blocks of approximately equal size. In a similar fashion as the tree-based subdivision scheme, this procedure is repeated until each subblock is either empty or consists of only a single pixel. An extension of this algorithm only splits down to 2×2 blocks and encodes their bit-patterns with Huffman coding [161].

Block-Coding with Prediction. Fränti and Nevalainen [97] extend hierarchical block coding by a prediction step that takes into account 2-D neighbourhoods of already encoded pixels. Based on static or adaptive probabilities of encountering a 1 for an observed neighbourhood configuration, newly encoded pixels are detected and the prediction error is stored instead. This reduces the entropy of the original. Furthermore, they apply reordering strategies that reduce the entropy further after the prediction and store the result with hierarchical block coding.

5.2.3 Quantisation

In addition to the positions of the inpainting mask, we also need to store the pixel values. An important tool for lossy compression of images is the reduction of brightness or colour values from a real-valued domain to a small number of integer values, the so-called *quantisation*. Since we deal with discrete, digital input images, the grey values of the ground truth are usually already quantised. In this thesis we assume input images with 8bit grey value depth. Thus, there are initially $2^8 = 256$ grey values $(0, ..., 255)$.

Reducing the number of grey values to some $q \in \mathbb{N}$, $q < 256$ is one of the easiest ways to reduce the file size. It has the added benefit that the human visual system can only distinguish a limited amount of grey values. Therefore, quantisation can be used for *perceptive coding* [280]. The examples in Figure 5.4 show that only for very coarse quantisations, the perceived quality deteriorates visibly.

In this thesis we perform only *uniform quantisation*. In order to reduce the initial number of $p \in \mathbb{N}$ grey values to a coarser quantisation with only $q < n$ values, we partition the grey value domain in q subintervals of length p/q. This defines a quantisation mapping of the original range $G_p = \{0, \ldots, p-1\}$ to the new grey value range $G_q = \{0, \ldots, q-1\}$: Every value from a given subinterval is mapped to the same grey value from G_q. Obviously, this mapping is lossy. A transformation of a value $x \in G_q$ back to the original dynamic range yields a *reconstruction value* y [268] from the original range G_p according to

$$y := \left\lfloor \frac{xp}{q} + \frac{1}{2} \right\rfloor . \tag{5.10}$$

Nevertheless, this backtransformation only yields q different values from G_p. Since this introduces an error to the known data, it is a lossy preprocessing step to the lossless entropy coding. In a PDE-based setting, the benefits of tonal optimisation can be diminished, if such a quantisation is applied afterwards. Therefore, in Section 5.3, we perform tonal optimisation under the constraint of the coarse quantisation.

Note that quantisation has a long tradition in signal processing and many more sophisticated quantisation techniques exist. Non-uniform quantisation allows to distribute quantised values over the full range of original values in such a way that a given error criterion is minimised. These ideas have already been pursued since the dawn of information theory (see e.g. Lloyd [173] and Max [191]). For image compression, virtually all lossy methods apply some kind of quantisation. For example, JPEG and JPEG2000 quantise transform coefficients in a non-

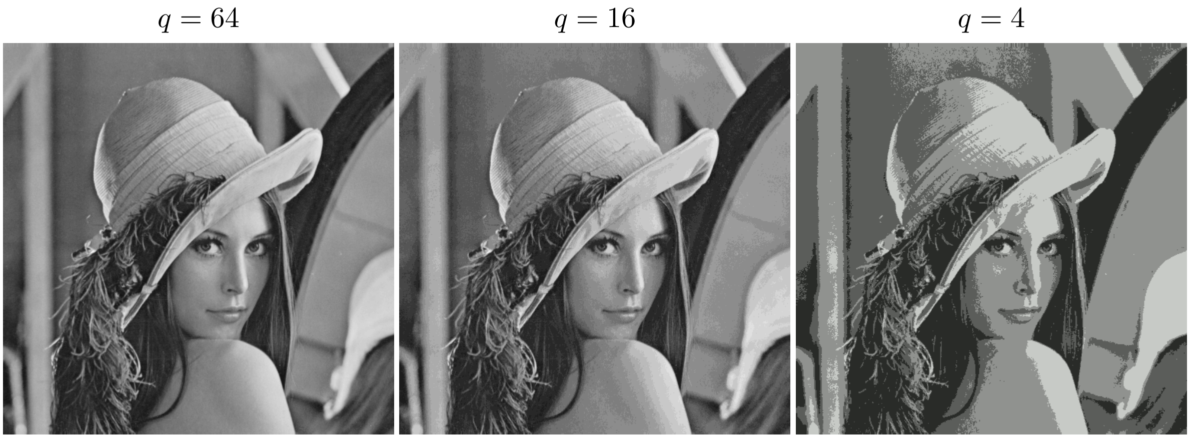


Figure 5.4.: **Quantisation of the test image *lena*.** The number q of different grey values is reduced from initially 64 (6 bit per pixel) to 16 (4 bit per pixel) and 4 (2 bit per pixel). Only at a very low quantisation level, humans perceive degradations, in particular in regions of soft transitions between different levels of brightness.

uniform way. In the case of colour images, one can even consider to quantise vectorial values directly. The drawback of these more complex methods is the need for additional optimisation and potential overhead, since the details of the non-uniform quantisation are needed for decompression. Gersho and Gray [105] provide a detailed overview of both scalar and vector quantisation.

5.3 Evaluating Inpainting Operators for Compression

> *Jealousy is the fear of comparison.*
>
> — **Max Frisch**
> (Writer)

Our goal is to evaluate three different diffusion-based inpainting operators with respect to their viability in compression: homogeneous, biharmonic, and edge-enhancing anisotropic inpainting. In particular, we are also interested in how the optimisation methods from Section 5.1 and the compression steps from Section 5.2 affect each operator.

Therefore, to assess the true potential of these operators, we have to design codecs that allow them to show their potential in a comparable setting. Note that the compression frameworks which we propose below work in a discrete setting. In the linear case, we consider the finite difference approximations of the inpainting equation in the same way as Mainberger et al. [184]. For EED, we use the standard discretisation (see Chapter 3.1) as in R-EED.

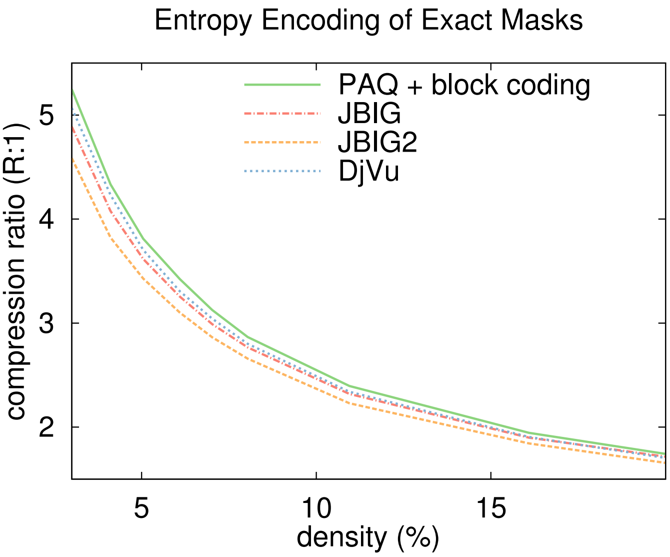


Figure 5.5.: **Coding of exact masks.** Comparison of different compression methods for exact masks obtained with an optimal control scheme for homogeneous inpainting of the image *peppers*.

We propose two different codecs depending on their selection and representation of known data: First we select exact (pixel-accurate) masks with optimal control schemes and stochastic sparsification in Section 5.3.1. Then we restrict ourselves to locally adaptive grids with an efficient tree representation in Section 5.3.2.

5.3.1 Exact Masks with Optimal Control

In the previous sections we have established all building blocks that are necessary to build a codec based on high quality, pixel-accurate masks. In particular, we discuss in the following, how we can store these masks efficiently.

We want our codec to have the following general structure: First, we find an optimal mask that contains a certain percentage of image points. This *mask density* acts as a quality parameter in the range 1 to 100, as in JPEG. We find this optimal mask by the algorithms from Section 5.1.1: for linear diffusion we use optimal control [59, 124] and for EED we employ stochastic sparsification with non-local pixel exchange [184]. This yields a binary image that needs to be stored. Furthermore, we have to decide how to integrate tonal optimisation and quantisation into our codec.

Storing Mask Locations. In order to store the binary image containing the locations of optimal known data efficiently, we have conducted an evaluation of the compression techniques from Section 5.2.2. We found that block coding schemes [97, 196, 320, 321] and coordinate coding [196] are outperformed significantly by encoders for binary images such as JBIG [146], JBIG2

[131] and DjVu [31]. However, they are viable as preprocessing steps for entropy coders such as PAQ [178].

Figure 5.5 shows compression experiments conducted on exact masks of different density. They were obtained with an optimal control scheme for homogeneous diffusion on the test image *peppers*. On first glance, the results are surprising: JBIG2 performs consistently worse than its predecessor JBIG, even though it features a more sophisticated pattern matching approach. However, one has to keep in mind that JBIG2 is designed for repeating patterns that occur for example in text from a scanned document. In contrast, our masks contain a lot of scattered points that make the creation of a dictionary with repeating patterns difficult.

Similarly, JBIG and DjVu are primarily designed for text documents and natural binary images that contain a lot of connectivity and regular patterns. This explains why the simple block coding scheme by Zeng and Ahmed [321] in combination with PAQ is the best choice. It reduces the file size by up to 10% in comparison to JBIG and DjVu depending on the mask density.

Storing and Optimising Grey Values. In order to store the grey value data associated with the mask efficiently, we have to choose the number q of quantised grey values. As mentioned in Section 5.2.3, performing such a coarse quantisation after tonal optimisation can affect the results negatively. The grey value optimisation algorithm from R-EED already considers the restriction to a set of coarse grey values. However, the least squares approaches for linear diffusion do not respect this constraint.

As a remedy, we propose *quantisation-aware grey value optimisation* in Algorithm 1. We can express the inpainting solution of the harmonic and biharmonic operator by a superposition of the inpainting echoes [184] from Section 5.1.2. For a given mask $\boldsymbol{c}$ and corresponding grey values $\boldsymbol{f}$, we denote the associated inpainting result from Section 5.1 as $\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})$. During optimisation, a Gauss-Seidel scheme successively updates the grey values at mask positions one by one. The crucial difference to the tonal optimisation algorithm of Mainberger et al. [184] is that we directly quantise the grey values after every update. Note that the most time-consuming part is the computation of the inpainting echoes. However, since the inpainting mask remains constant, the echoes can be reused for arbitrary quantisation parameters q. Therefore, we are able to optimise q thoroughly and efficiently for the linear operators. For EED, this is more costly.

The choice of q influences the overall file size, since the entropy coding of the grey values becomes more efficient for smaller numbers of different grey values. Therefore, decreasing the parameter q also reduces the file size in general. Simultaneously, the error increases, since the optimised grey values are misrepresented. This negative effect can be somewhat attenuated by the quantisation aware grey value optimisation, but is still present.

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

Algorithm 1: Quantisation-aware grey value optimisation.

Consequently, for a given mask, a suitable parameter q must be found that offers the best trade-off between file size and reconstruction quality. This means that both the inpainting error and the file size have to be minimised simultaneously. For a given quantisation parameter $q \in \{0, ..., 255\}$, let $s : \{0, ..., 255\} \to \mathbb{N}$ be the file size in byte and $e : \{0, ..., 255\} \to \mathbb{R}$ the corresponding mean square error. By normalising both quantities to the range $[0, 1]$ and combining them additively, we define the *trade-off coefficient* μ as

$$\mu := \frac{s(q)}{s(255)} + \frac{e(q)}{e(255)}. \tag{5.11}$$

The smaller this coefficient, the better the trade-off for a given q. Our goal is to find the best q for a given mask. To this end, we minimise μ with respect to q in combination with quantisation-aware grey value optimisation. This implies a three-step codec for exact masks:

1. Select a fraction d of total pixels with the optimal control approach [124] (harmonic/biharmonic) or stochastic sparsification [184] (EED).

2. Perform quantisation-aware grey value optimisation and select the quantisation parameter q with optimal trade-off between file-size and reconstruction quality.

3. Optimise block size for optimal compression with PAQ. Concatenate header, positional, and grey value data and apply PAQ to the total file.

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

Algorithm 2: Stochastic tree densification.

The reconstruction is straightforward. All the entropy-coded data is recovered and a single inpainting reconstructs the image.

5.3.2 Stochastic Tree-Building

In this section, we pursue an approach that restricts known data to a regular adaptive grid. In order to lower the coding cost of these locations, we use the *binary subdivision tree* representation from Section 5.1.1. Unfortunately, there is no straightforward extension of the optimal control approaches for exact masks from the previous section. Moreover, only the heuristic subdivision scheme from R-EED [252] has been used so far to obtain tree-based masks. For a fair comparison, we want to stay as close to the exact codec as possible with respect to the optimisation strategies. Therefore, we extend the stochastic approach of Mainberger et al. [184] to subdivision trees in the following.

If we want to transfer the basic concepts of stochastic sparsification from Section 5.1.1 to a binary tree representation, there are some key differences: We have experimentally determined that densification is more efficient for tree structures than sparsification. Therefore, we start with a small amount of data and iteratively add more points at locations with large error until the target density is reached.

In addition, we consider to add nodes to the tree instead of dealing with mask points directly. Since we want to perform a single additional subdivision, the tree structure dictates that only subimages corresponding to leaf nodes may be split. Such a split is equivalent to adding two

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

Algorithm 3: Nonlocal node exchange.

child nodes to the leaf node (see Figure 5.2). Note that several mask points might be added by a single subdivision (the corners and the midpoint of the corresponding subimage). These mask points might also be contained in several of the neighbouring subimages.

Furthermore, the error computation must be adapted. In order to avoid a distortion of the influence of each node, we do not consider the mean squared error in each subimage, but the sum $e(t_k)$ of squared differences

$$e(t_k) = \sum_{(i,j)\in\Omega_k} (f_{i,j} - u_{i,j})^2 \tag{5.12}$$

where Ω_k denotes the image domain of the subimage corresponding to the tree node t_k. This avoids that the same per-pixel-error in small subimages is weighted higher than in large subimages. Taking all these differences into account, we define *stochastic tree densification* in Algorithm 2. For a target density d, it produces an optimised tree T with a corresponding pixel mask $C(T) \subset \Omega$.

Just as for the original sparsification approach, there is a risk that Algorithm 2 is trapped in a local minimum. To avoid this problem, we propose Algorithm 3, an adapted version of the nonlocal pixel exchange (NLPE) of Mainberger et al. [184] that we have described in Section 5.1: They first remove random points from the inpainting mask. Then they replace them with potentially better non-mask pixels.

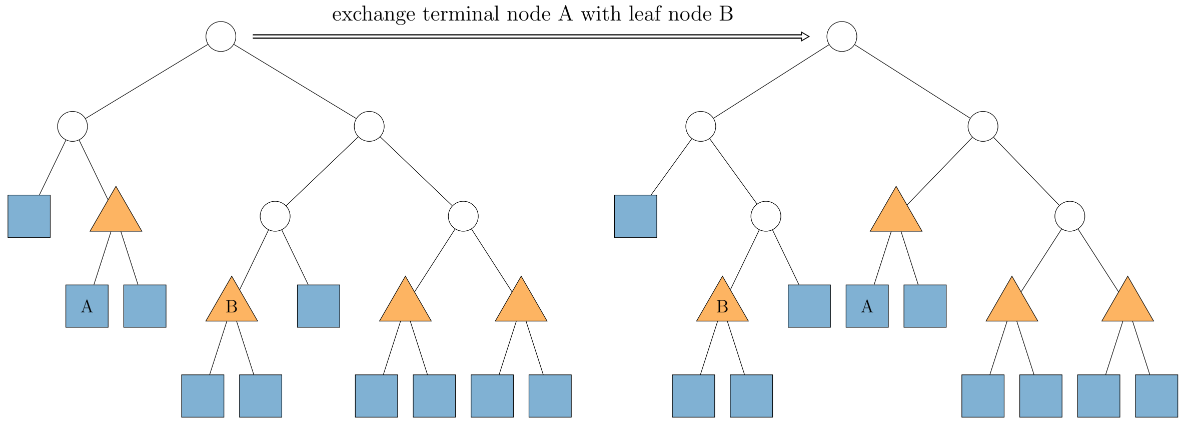


Figure 5.6.: **Nonlocal Node Exchange.** This Figure visualises the concept of exchanging terminal and leaf nodes. Terminal nodes are not split any further, which means that both of their children are leaf nodes. Leaf nodes are marked as blue rectangles, terminal nodes as orange triangles. A single exchange corresponds to reversing the split of a terminal node and splitting a leaf node instead. In our example, we exchange the position of the leaf node A and the terminal node B.

In the following we transfer this concept to our subdivision trees. Most importantly, we have to respect the tree structure in order to define a *nonlocal node exchange (NLNE)*. In the first step, we want to remove n randomly selected nodes from the tree. However, this set underlies some restrictions: It can only consist of nodes that are split exactly once. This is the case if and only if both children of a node are leaf nodes. We call these nodes *terminal nodes*. The reversion of their associated image split comes down to removing their leaf nodes. Thereby, we convert the terminal node to a leaf node.

In the second step of the NLNE, we want to add back nodes to the tree. First, we select a candidate set of m leaf nodes. From these candidates, we select the n nodes that correspond to the subimages with the highest reconstruction error w.r.t. the initial mask. We split these subimages by adding both children to the tree, thus converting leaf nodes into terminal nodes.

These modifications lead to Algorithm 3. The example in Figure 5.6 illustrates that the modifications applied to the tree by our algorithm can be interpreted as swapping the positions of pairs consisting of a leaf node and a terminal node.

Finally, the binary trees obtained from the densification and nonlocal node exchange can be stored as a sequence of bits. As in the example from Figure 5.2, we store a maximum and minimum tree depth and only save the node-structure explicitly inbetween. The only additional required header data are the image size and the number q of quantised grey values. We combine the tree densification with the same strategies for grey-value optimisation and quantisation as in the previous section and obtain the following four-step compression pipeline:

1. Select a fraction d of total pixels with tree densification (Algorithm 2).
2. Optimise the splitting tree with nonlocal node exchange (Algorithm 3).
3. Perform grey value optimisation and optimise the quantisation parameter w.r.t. the trade-off between file size and reconstruction quality.
4. Concatenate header, positional, and grey value data and apply PAQ.

For reconstruction, we decode the PAQ container, extract positional data from the tree, combine it with the grey values and perform a single inpainting.

5.4 Experiments

> *It doesn't matter how beautiful your theory is,*
> *it doesn't matter how smart you are.*
> *If it doesn't agree with experiment, it's wrong.*
>
> — **Richard P. Feynman**
> (Physicist)

In the following we evaluate the capabilities of harmonic, biharmonic, and EED inpainting for the two compression methods from the previous sections. Our experiments rely on a set of widely used test images. First, we evaluate the sensitivity of the individual operators under different optimisation and compression steps. Then we compare their overall performance to R-EED, which marks the current state of the art in PDE-based compression, and to the transform-based coders JPEG and JPEG2000.

Influence of Data Selection Strategies. In Figure 5.8 we compare inpainting results with three different masks that contain 5% known data of the 256×256 image *peppers*: A random mask containing the same uniformly distributed locations for all three algorithms, an exact mask obtained with optimal control or stochastic sparsification, and a restricted mask from tree densification with non-local node exchange (see Figure 5.7). The optimised masks are different for each inpainting operator and we do not apply coarse quantisation. For all three methods we have performed tonal optimisation.

EED shows to be far less sensitive to the restriction to an adaptive grid and provides the overall best reconstruction quality for all cases. Biharmonic inpainting performs better than harmonic inpainting in general and also less sensitive to tree-based known data. Also, the results for biharmonic inpainting are more visually pleasing in general: The smoothness constraints of the biharmonic operator avoid the typical singularities that occur at known data for harmonic inpainting. Hoffmann et al. [128] have explained this phenomenon in terms of

Green's functions. These singularities are particularly obvious in Figure 5.8 (a), the harmonic inpainting with random known data. EED also avoids these singularities and is able to produce the best result consistently. Especially its robust performance on the suboptimal tree-based grid is remarkable. However, it should also be noted that for exact masks, the difference between the three diffusion models is small and would not justify to use the complex nonlinear model instead of the efficient linear ones.

Influence of Mask Density. In order to asses the influence of the mask density on the inpainting results, we optimise exact and restricted masks with different densities, perform grey-value optimisation and compare the mean square error (MSE) at the same mask density. The results in Fig. 5.9 (a) show that, in general, biharmonic performs better than harmonic inpainting given the same amount of known data. This is consistent with previous results [59]. EED can outperform both methods, but the difference is only significant for low densities. Note, however, that sparse masks are exactly the requirement for efficient compression. The graph also shows again that the restriction of the mask to an adaptive grid has a significant negative impact on the quality. This affects harmonic inpainting more than its biharmonic counterpart and EED.

Influence of Quantisation. Some interesting and surprising results come from a comparison w.r.t. the influence of quantisation. In Figure 5.10 we compare results with exact masks and grey value optimisation. This time, we apply a coarse quantisation to 64, 32, and 16 individual grey values. Interestingly, the grey value optimisation is able to compensate for this negative effect very well in the case of harmonic diffusion and EED: The increase of the error is almost negligible compared to the results without quantisation from Figure 5.8. However, the higher-order biharmonic inpainting suffers a lot more. As we will see in the following, this affects compression performance significantly.

Compression Performance. An evaluation of the actual compression performance with the codecs from Sections 5.3.1 and 5.3.2 in Fig. 5.9(b) shows a significantly different ranking than in the density comparison. For exact masks, harmonic inpainting can even surpass its biharmonic counterpart. The coding cost for the known data is similar in both cases, but since harmonic inpainting is less sensitive to a coarse quantisation of the grey values, it performs overall better than biharmonic inpainting. The drawbacks of the restrictions in the tree-based approach are attenuated by the reduced positional coding cost. After a break-even point around ratio 20:1, the biharmonic tree-based method outperforms both exact approaches. Since EED does not have distinct advantages at high mask densities, it does not outperform the linear methods at low compression ratios. At high compression ratios however, its robustness under both restricted locations and coarsely quantised grey values allow it to outperform its competitors.

Comparison to Other Encoders. In relation to transform-based coders, the tree-based method performs consistently better than JPEG and in many cases also outperforms JPEG2000 for

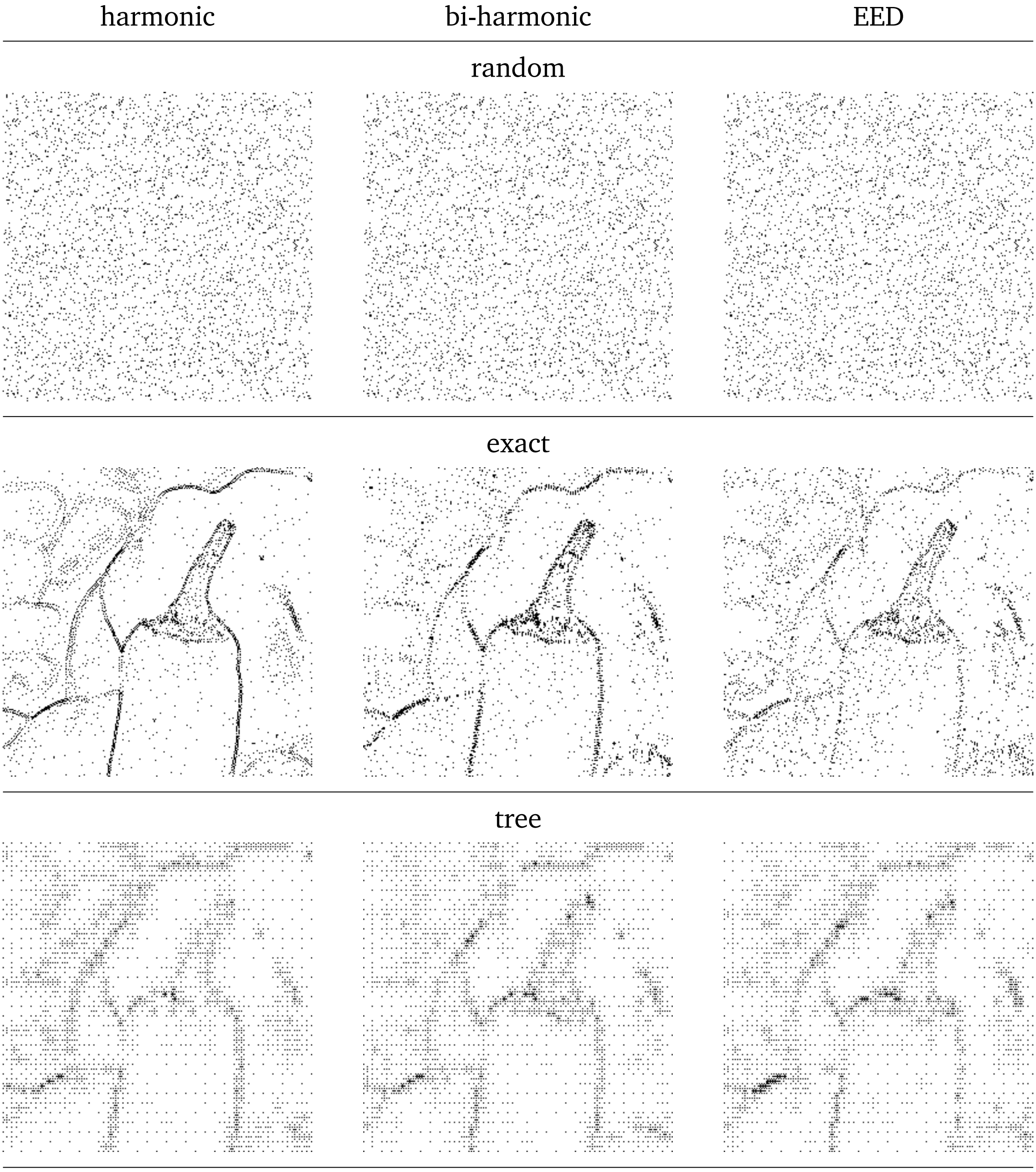


Figure 5.7.: **Optimal Inpainting Masks for Different PDEs.** All masks contain 5% of the total image points of the test image *peppers*. The locations of known data are marked in black. **(a) Top row:** The same uniformly distributed random mask is used for all three diffusion types. **(b) Middle row:** For the harmonic mask, points are stored left and right of edges. In the biharmonic case, the structure remains similar, but points either spread out more or cluster closer together to store a whole region verbatim. The EED mask is spread out much more. **(c) Bottom row:** The masks from tree densification with nonlocal node exchange follow a similar pattern like the exact ones. However, the differences between the choice of locations is less pronounced than in the exact case due to the reduced number of possible choices.

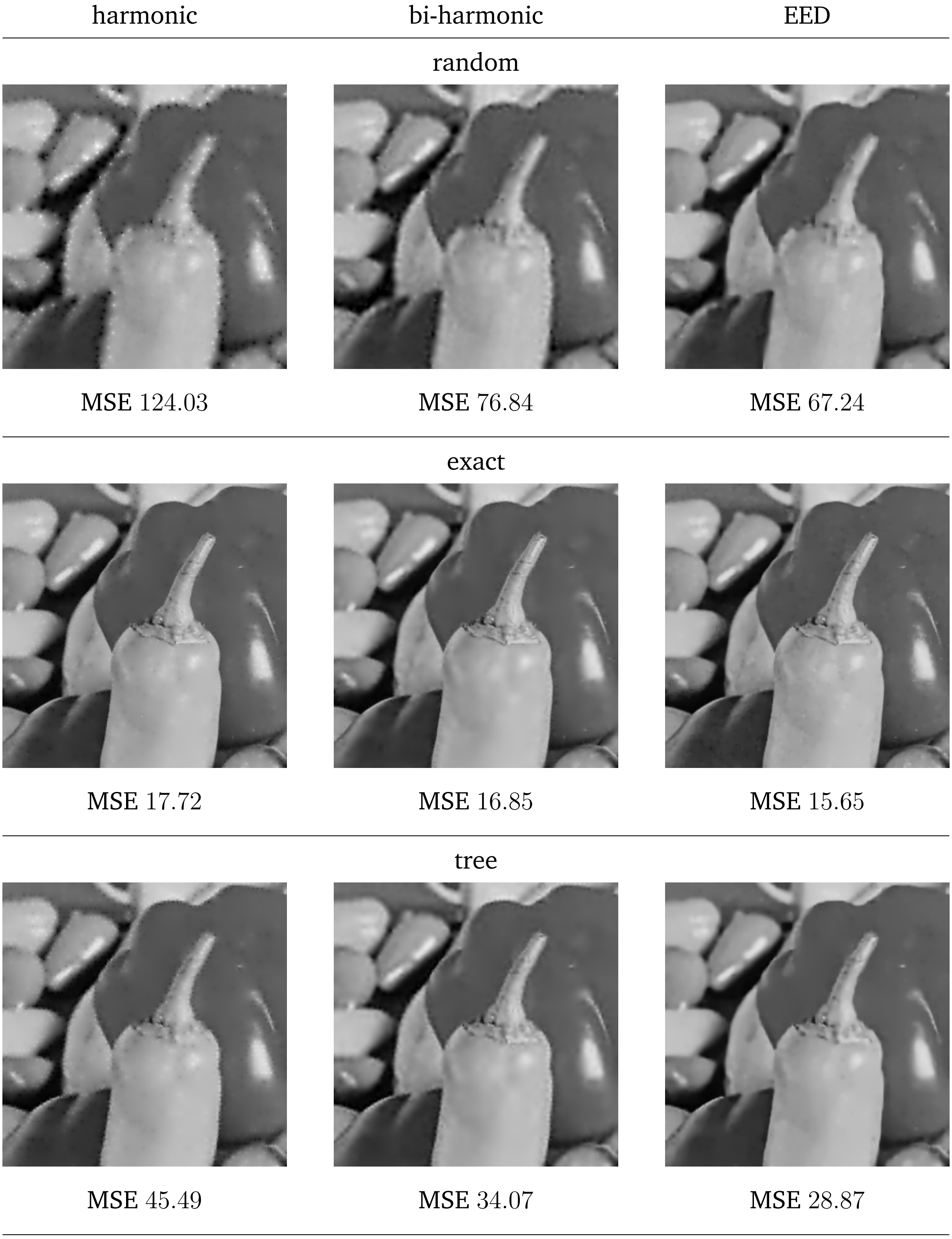


Figure 5.8.: **Influence of Data Selection Strategies on Inpainting.** Reconstruction from the 5% masks from Figure 5.7 with different inpainting operators. **(a) Top row:** Both biharmonic and EED perform much better than harmonic on random data. This already indicates their higher robustness to suboptimal known locations. Moreover, harmonic suffers from severe singularities. **(b) Middle row:** The difference between the three inpainting operators is much less pronounced for optimal spatial and tonal data. The singularities are still there for harmonic inpainting, but they are hardly visible in print. **(c) Bottom row:** Harmonic inpainting suffers the most from the restriction to a locally adaptive grid. Biharmonic and EED lose less quality compared to optimal locations.

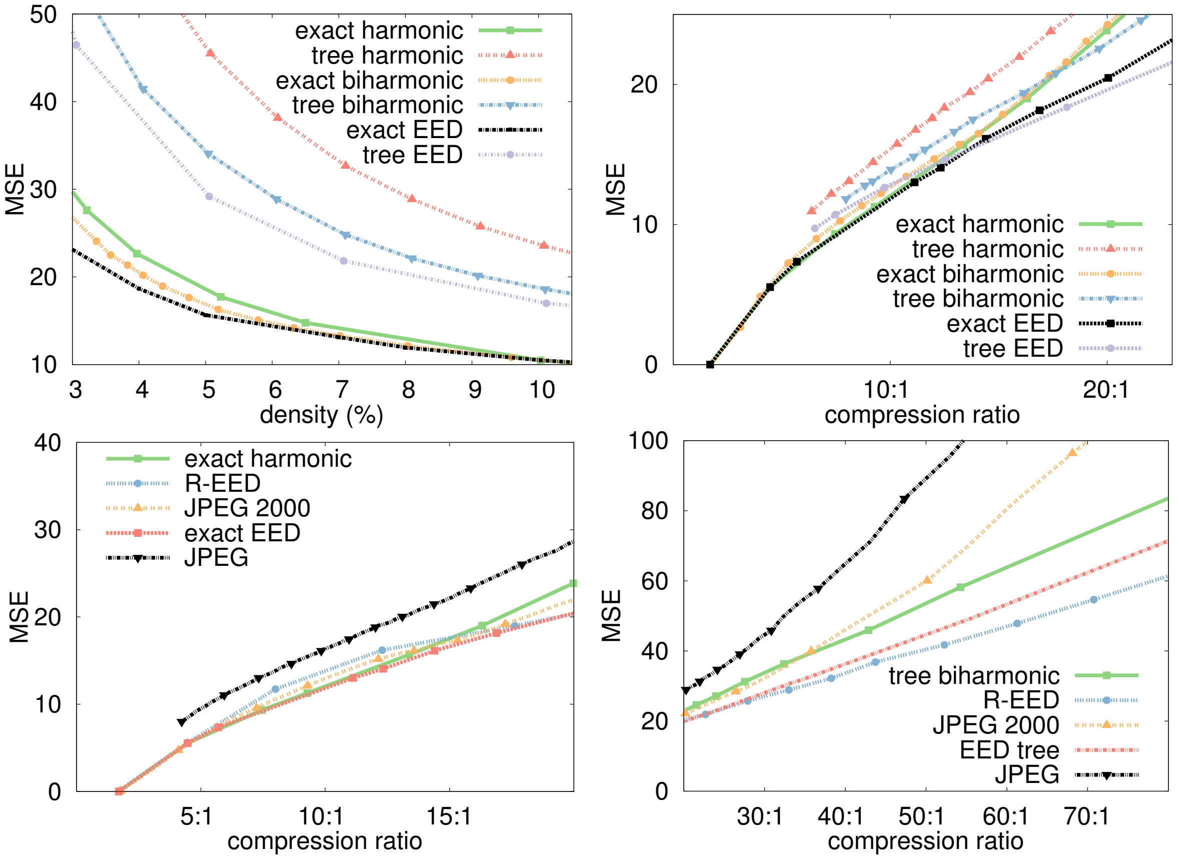


Figure 5.9.: **Comparisons for the 256×256 Image *peppers*.** The top row compares harmonic and biharmonic versions of our codecs, and the bottom row compares our best methods to transform coders and R-EED. **(a) Top Left:** Comparison at same mask density. **(b) Top Right:** Comparison at same compression ratio. **(c) Bottom Left:** Low to medium compression ratios. **(d) Bottom Right:** High compression ratios.

compression ratios larger than 35:1. Surprisingly, for very high compression ratios, the heuristic approach of R-EED outperforms the more sophisticated stochastic tree densification with EED. While this seems counter-intuitive at first glance, there is a simple explanation: R-EED treats entropy coding and quantisation in a different fashion. Our method first selects a single tree and then optimises q for the best trade-off between file size and reconstruction quality. In contrast, R-EED defines a target compression ratio first and then builds a lot of different trees that fit to this ratio. Thus, it already incorporates coding costs into the selection of known data, not afterwards. At low compression ratios, this does not have a high impact, but it makes a difference for very small files.

In comparison to other PDE-based methods, linear diffusion performs best in the area of low to medium compression ratios (up to 15:1). Fig. 5.9 shows that it can beat both R-EED and JPEG2000. On smooth images like *peppers*, harmonic diffusion with exact masks even outperforms JPEG2000. This demonstrates how powerful simple PDEs can be.

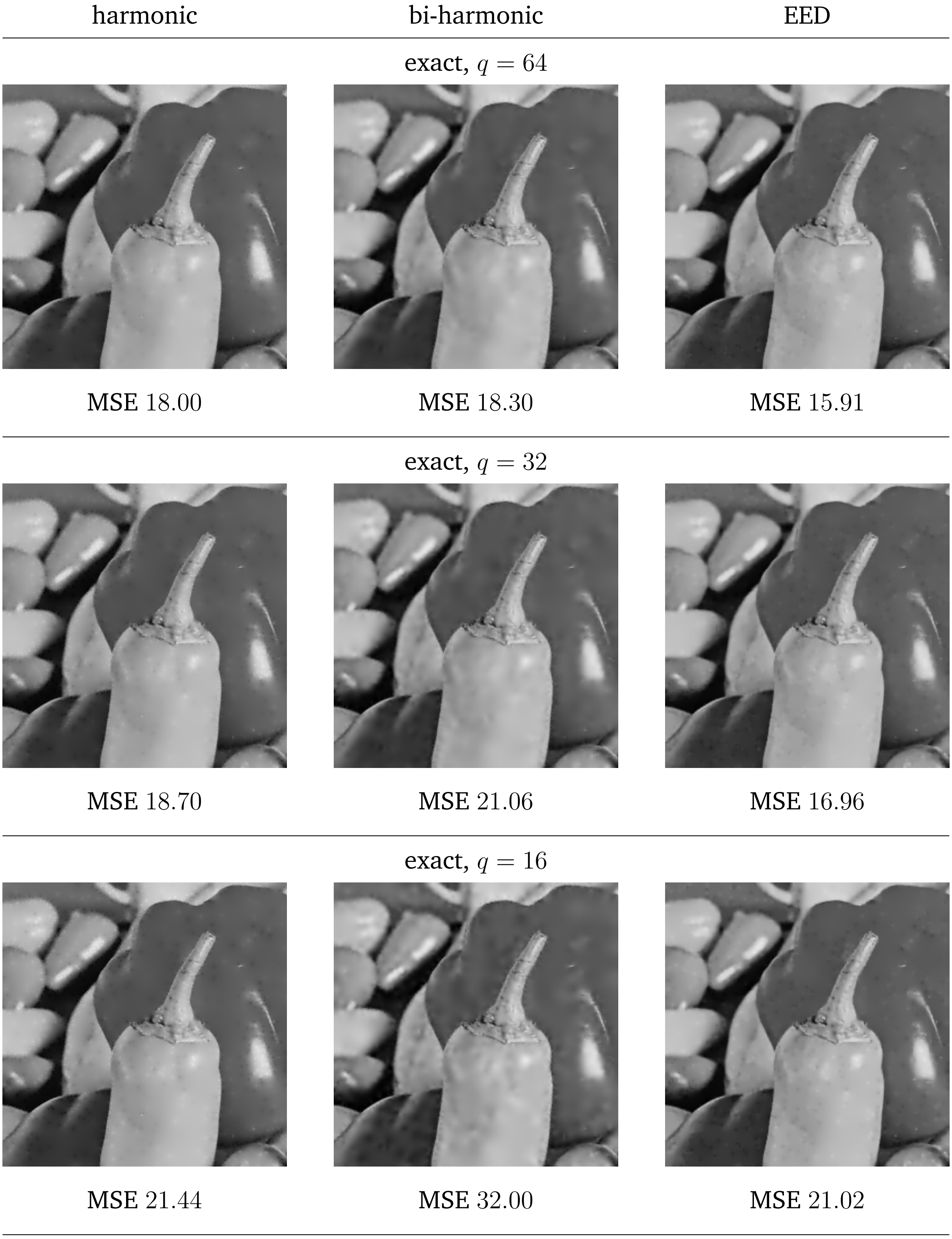


Figure 5.10.: **Influence of Quantisation on Inpainting.** Reconstruction from the exact 5% masks from Figure 5.7 with different inpainting operators. For all results, we have performed quantisation-aware grey value optimisation with 64, 32, and 16 different grey values. Reducing the number of grey values from 256 to 64 does not change the quality dramatically for all diffusion methods. However, the ranking of harmonic and biharmonic inpainting has already changed compared to Figure 5.8. Reducing the parameter q even further reveals that harmonic and EED inpainting are much less sensitive to quantisation than biharmonic interpolation.

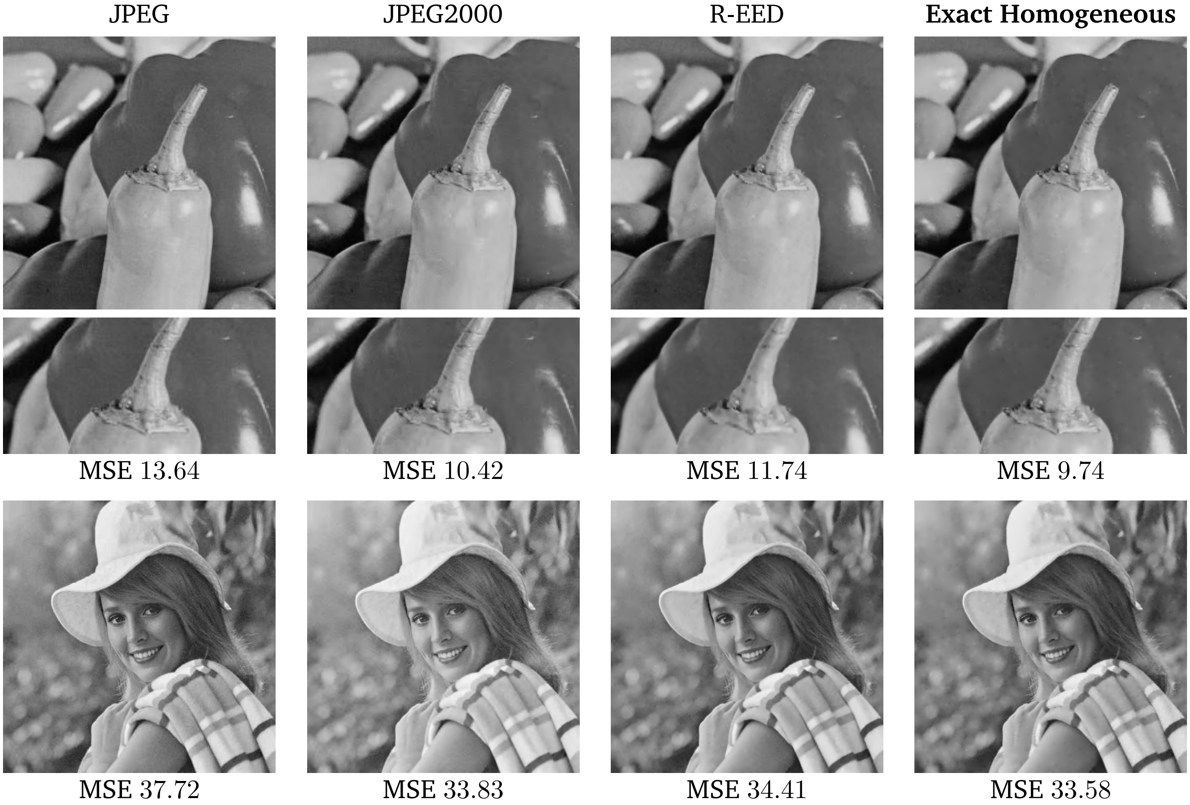


Figure 5.11.: **Compression Results for *peppers* and *elaine*. (a) Top:** Results for *peppers* (256×256 pixels) with compression ratio $\approx$ 8:1. **(b) Bottom:** Results for *elaine* (512×512 pixels) with compression ratio $\approx 18 : 1$.

In Tab. 5.1, we allow both edge-enhancing anisotropic diffusion with probabilistic tree densification and exact harmonic masks obtained with optimal control. On low compression ratios, all of our results outperform or match R-EED and are highly competitive to the quality of JPEG2000. Regarding high compression ratios, the performance is comparable to R-EED. However, the more efficient integration of the entropy coder into the choice of mask points gives R-EED still a slight edge on some images.

Table 5.1.: MSE comparison on several test images. For the compression ratio of 15:1 we use exact masks with homogeneous inpainting for elaine and lena. The rest of the error values for our method are obtained with tree-based EED.

Ratio	$\approx 15 : 1$				$\approx 60 : 1$			
Image	elaine	lena	trui	walter	elaine	lena	trui	walter
JPEG	34.69	20.06	16.24	6.73	77.39	97.37	116.98	69.40
JPEG2000	**31.23**	**14.73**	12.27	5.70	54.18	60.48	88.96	47.09
R-EED	35.48	16.56	11.27	5.48	**49.87**	56.96	**44.38**	24.05
Our Method	31.38	17.00	**10.48**	**4.53**	54.83	**53.56**	47.42	**21.74**

5.5 Conclusion

> *People forget how fast you did a job,*
> *but they remember how well you did it.*
>
> — **Howard W. Newton**
> (Advertising Executive)

The investigation of the interplay between diffusion-based inpainting and different compression steps in this chapter has both practical implications and provides general insights into codec design. On the practical side, we have shown that codecs with parameter-free linear inpainting PDEs can beat both the quasi standard JPEG2000 of transform-based compression and the state of the art in PDE-based compression. This is an indication that simple and fast PDEs have a lot of potential that has not been fully used to this point. In particular, they might be integral for the design of *synchronous* compression codecs that are fast in both compression and decompression. However, the selection of known data still hinders this goal: Optimal control and stochastic sparsification have not yet reached real-time performance, but linear diffusion needs thorough optimisation to produce competitive quality. Therefore, future research should focus on faster mask selection algorithms.

A valuable general insight from this chapter concerns the comparison of inpainting operators: *The performance of PDEs for compression can only be evaluated in the context of actual codecs.* Comparisons that do not consider all compression steps can lead to false rankings of inpainting operators that do not reflect their real compression capabilities. In particular, the sensitivity of the biharmonic operator to coarsely quantised known data makes the simpler harmonic diffusion the preferable choice for compression. Finally, there are some concrete implications for the codec design in the remaining chapters of this thesis. Together with the statistical justification in Chapter 4, our comparison of diffusion operators in a compression context has shown that EED is the best choice, if mask positions are used that are suboptimal, but cheap to store. Therefore, we pursue a combination of subdivision trees and EED for our investigations on textured images, real-time decompression, and perceptive coding.

Publication Info

The exact and tree-based framework for linear diffusion was first presented by Peter, Hoffmann, Hoeltgen, and Weickert [223] at the Pacific Rim Symposium 2015. This chapter corresponds to a technical report by the same authors that also includes EED [222]. Preliminary versions of the stochastic tree densification were developed by Frank Nedwed [204] in his Bachelor's thesis under the supervision of Pascal Peter.

6 Consistency: Combining Diffusion- and Exemplar-based Inpainting

> *Opposites are not contradictory but complementary.*
>
> — **Niels Bohr**
>
> (Physicist, Winner of the Nobel Prize)

As we have already discussed in the previous chapters, the current generation of PDE-based encoders is competitive to the state of the art in transform-based compression. In particular, the R-EED codec of Schmaltz et al. [250] can surpass JPEG2000 quality-wise. However, we have only considered images with low or moderate amounts of texture such as *peppers* or *elaine*, so far.

In such cases, the key element to the success of diffusion-based codecs is the ability of edge-enhancing anisotropic diffusion (EED) [295] to reconstruct images from sparse pixel data. It is important to note here that EED inherently imposes a smoothness assumption on the missing image parts. Due to its ability to preserve sharp edges, coarse structure can be accurately reconstructed. However, for fine-scale textures, the smoothness assumption is violated at many locations. This means that either large amounts of known data are necessary to store them or they simply cannot be reconstructed. Consequently, the performance of diffusion-based algorithms degrades with increasing amount of texture content in the original images. In such cases, transform-based coders still provide superior results.

However, there are other reconstruction methods that are better suited for the interpolation of texture from partially known data. In 1999, Efros and Leung [84] pioneered exemplar-based inpainting for the purpose of extending images and filling in missing or corrupted image parts. More recently, Facciolo et al. [90] have proposed exemplar-based inpainting methods that are suited for sparse known data. In the following, we explore the potential of this sparse exemplar-based inpainting for image compression.

Our Contribution. We assess the suitability of exemplar-based inpainting for image compression and compare it to diffusion-based inpainting. Following the results of this analysis, we construct a novel compression codec that combines the strengths of both approaches while minimising the effect of their drawbacks. In our experiments on well-known test images we demonstrate that this hybrid inpainting approach can beat established diffusion-based methods and is competitive to transform codecs also for images with rich texture.

Related Work. In Chapter 5 we have concluded that the currently most promising strategy for compression uses subdivision to locate and store the positions of known data efficiently in terms of a binary tree. Therefore, we rely on the anisotropic R-EED approach by Schmaltz et al. [250, 252] as the foundation of our method. Chapter 3.3 contains a more detailed description of the R-EED codec. While our novel codec is the only diffusion-based method that deals specifically with textured images, there are PDE-based coders dedicated to other classes of data such as cartoon images [183], 3-D data [218], or depth maps [103, 127, 166]. For a more detailed general overview of these codecs we refer to Chapter 3.2.3.

The patch-based inpainting on sparse images by Facciolo et al. [90] that we use in our new method is related to a long line of classic exemplar-based approaches. The underlying idea of synthesising patterns dates back to the dawn of computer science. For example, Shannon and Weaver [260] tried to approximate the English language in terms of recurring patterns of letters. In image processing, similar ideas are pursued to reconstruct textures. Exemplar-based texture synthesis was made popular by the influential work of Efros and Leung [84]. Since a full review of the field is beyond the scope of this thesis, we focus on selected publications that are related to our own work and refer to Arias et al. [10] as well as Guillemot and Le Meur [116] for in-depth reviews.

During the last decade, the concept of combining structure adaptive inpainting with exemplar-based ideas has been explored in several different directions. The approach of Bertalmío et al. [27] comes closest to our method since it also employs an explicit decomposition into a cartoon and a texture image. The cartoon reconstruction relies on an inpainting process that propagates information along isophotes. Patch-based inpainting restores missing parts of the texture image. In contrast to our codec, they use an additive decomposition. This doubles the amount of original data, which is disadvantageous for compression.

Many patch-based approaches incorporate the image structure as additional guidance information. For instance, Drori et al. [80] propose a multi-scale approach that guides texture inpainting by smooth image structures. Sun et al. [272] perform patch-based texture reconstruction along manually specified curves, while Criminisi et al. [67] prioritise the reconstruction of missing image points in such a way that existing image structures are continued. In the work of Cao et al. [44], level lines extracted from a simplified version of the image are the guidance feature for exemplar-based inpainting. A different approach is pursued by Arias et al. [10] who include gradient information in a variational model for exemplar-based inpainting.

Some texture synthesis approaches also rely on learning. For instance, Hees et al. [122] modify the Field-of-Experts (FoE) approach of Roth and Black [238] (see Chapter 4) with new potential functions that allow to learn synthesis filters from a given database of textures. Similarly, Yu et al. [318] have shown that learning methods for multi-stage diffusion models [58] can be used on the known data of a partially given image to reconstruct missing texture.

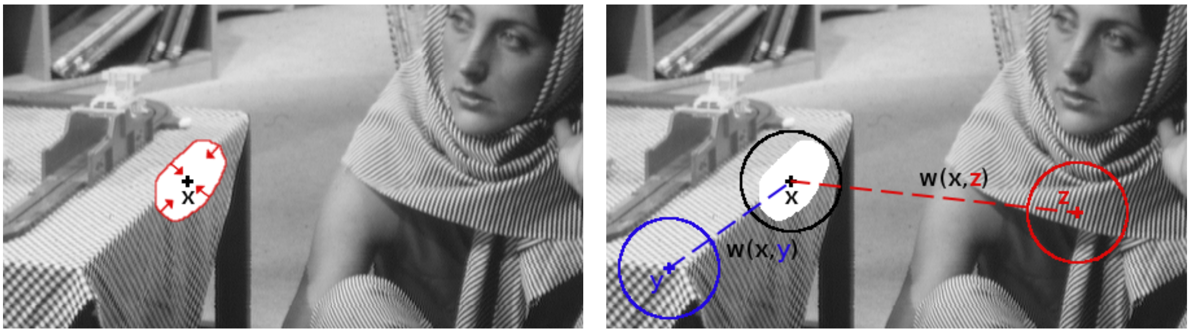
All aforementioned publications focus on image inpainting. In regards to actual compression, there are related approaches that modify existing transform-based coders with exemplar-based inpainting. Rane et al. [232] propose a scheme that removes selected JPEG blocks and reconstructs them either with the method of Efros and Leung [84] or structure inpainting. The method of Liu et al. [170] focuses on removing visual redundancy in transform coders like H.264 or JPEG. They aim at human perception and do not necessarily minimise the quantitative error. To this end, the image is decomposed into edge-regions that are reconstructed with a combination of structure propagation and exemplar-based inpainting and texture regions that are synthesised with purely patch-based methods. Xiong et al. [312] consider a similar decomposition, but they focus on inpainting of homogeneous areas with a custom PDE-based approach. They do not address texture synthesis and use JPEG in the remaining regions.

Moreover, there are distantly related methods from the area of compressed sensing. In this area of signal processing, researchers aim to reconstruct signals from small amounts of known data and prior knowledge. These methods can be interpreted as dictionary approaches (e.g. [7, 85, 120]) and use databases of image prototypes or patches for reconstruction instead of relying on partially known data like the exemplar-based methods. Aharon et al. [7] also explicitly propose compression as one application of their approach.

Organisation of the Chapter. First we explain the concept of the exemplar-based inpainting technique of Facciolo et al. [90] in Section 6.1. In Section 6.2 we review diffusion inpainting very briefly and assess the strengths and weaknesses of both approaches with respect to image compression. From these conclusions we motivate a novel hybrid compression scheme in Section 6.3 and analyse its performance in Section 6.4. We conclude the chapter with a summary and outlook on future work in Section 6.5.

☞ **Texture in Colour Images**

In this chapter we only deal with grey scale images. However, our colour codec from Chapter 8.2 also helps to compress texture, even though it is not specifically designed for this purpose. It decomposes the image into brightness and colour information and allows to store structure in the brightness channel very accurately by assigning it a larger file size budget. Note that this channel could also be compressed with our hybrid algorithm.



(a) diffusion propagation

(b) patch-based propagation

Figure 6.1.: **Propagation with Diffusion-and Exemplar-based Inpainting.** Consider a missing textured area in the image *barbara*. Diffusion based inpainting (a) propagates structural information continuously from the boundary of the patch inwards. This propagation allows to produce edges, but not jumps, which makes it impossible to reconstruct chequered patterns. In contrast, patch-based propagation (b) averages the centre pixels of all other image pixels with weights $w(\cdot,\cdot)$ that are larger for similar patches. They can thereby also recreate regular patterns. For concrete inpainting results see Figure 6.3.

6.1 Exemplar-based Inpainting

> *Example is the best precept.*
>
> — **Aesop**
> (Poet)

The so-called *nonlocal inpainting (NLI)* approach of Facciolo et al. [90] follows the core idea of all patch-based methods: Missing information is filled in by exchanging information between image patches. However, in contrast to other algorithms from the field, it allows inpainting from sparse data. Therefore, we can compare it to the inpainting capabilities of EED from Chapter 3.1. For the sake of comprehensibility, we only discuss a special case of the flexible NLI framework: We have chosen algorithm AB with patch-wise non-local means from [90], since the experiments by Facciolo et al. [90] suggest that it is best-suited for the specific task at hand.

Let us consider the same inpainting problem as in the previous chapters. For a grey scale image $f : \Omega \mapsto \mathbb{R}$ with a rectangular image domain Ω, we want to find a reconstruction u of the missing data on the *inpainting domain* $\Omega \setminus K$ from the sparse known data on the *inpainting mask* $K \subset \Omega$. In essence, NLI reconstructs u by minimising a patch similarity function V between pairs of image patches. It forces unknown pixel values in one patch to be similar to known values in the other patch. To this end, consider two disk-shaped patches centred in image points $\boldsymbol{x}$ and $\boldsymbol{x}'$, respectively. The similarity function V is defined as the weighted squared difference

$$V(\boldsymbol{x},\boldsymbol{x}') = \int_D g(\boldsymbol{x},\boldsymbol{x}',\boldsymbol{y})\Big(u(\boldsymbol{x}+\boldsymbol{y}) - u(\boldsymbol{x}'+\boldsymbol{y})\Big)^2 \, d\boldsymbol{y}. \tag{6.1}$$

Here, D is a disk around the origin, and $\boldsymbol{y}$ a coordinate relative to the respective patch centre. A common practice in patch-based methods is to rescale the individual pixel differences $u(\boldsymbol{x}+\boldsymbol{y})-u(\boldsymbol{x}'+\boldsymbol{y})$ with Gaussian weights that reflect descending importance with the distance to the patch centres. However, in NLI, the weights g additionally account for the fact that, given a sparse inpainting mask, both patches can contain similar amounts of known data. Thus, a mutual exchange of information can be beneficial. The weights are defined as

$$g(\boldsymbol{x},\boldsymbol{x}',\boldsymbol{y}) \quad = \quad \frac{K_\sigma(\boldsymbol{y})}{\rho(\boldsymbol{x},\boldsymbol{x}')}(\chi_K(\boldsymbol{x}+\boldsymbol{y}) + \chi_K(\boldsymbol{x}'+\boldsymbol{y})), \tag{6.2}$$

where $\rho(\boldsymbol{x},\boldsymbol{x}')$ is a normalisation term that ensures $\int_D g(\boldsymbol{x},\boldsymbol{x}',\boldsymbol{y})d\boldsymbol{y} = 1$ and K_σ is a Gaussian with standard deviation σ. The characteristic function χ_K of the set of known data indicates where similarities between patches should be enforced. Let $\boldsymbol{x}_1 := \boldsymbol{x}+\boldsymbol{y}$ denote an image point from the first patch and $\boldsymbol{x}_2 := \boldsymbol{x}'+\boldsymbol{y}$ the corresponding point in the second patch. If both points are unknown, i.e. not contained in K, g becomes 0 and thus, no information exchange takes place. If at least one of the two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is known, we have $g > 0$ and thus V enforces similarity between those two pixels.

The second important ingredient of NLI is the decision, for which pairs of patches the similarity function V should be minimised. To this end, Facciolo et al. introduce a patch similarity weight function w and minimise the energy

$$E(u,w) = \frac{1}{h}\int_\Omega\int_K w(\boldsymbol{x},\boldsymbol{x}')V(\boldsymbol{x},\boldsymbol{x}')\,d\boldsymbol{x}\,d\boldsymbol{x}' - \int_\Omega H(\boldsymbol{x},w)\,d\boldsymbol{x}, \tag{6.3}$$

$$\text{s.t.} \quad \int_K w(\boldsymbol{x},\boldsymbol{x}')\,d\boldsymbol{x}' = 1. \tag{6.4}$$

Optimal weights w minimise the weighted total patch error according to V while maximising the entropy

$$H(\boldsymbol{x},w) = -\int_K w(\boldsymbol{x},\boldsymbol{x}')\log w(\boldsymbol{x},\boldsymbol{x}')\,d\boldsymbol{x}'. \tag{6.5}$$

For a given u, the patch similarity weights w impose a Gaussian-like weighting of the patch differences $V(\boldsymbol{x},\boldsymbol{x}')$:

$$w(\boldsymbol{x},\boldsymbol{x}') = \exp\Big(-\frac{1}{h}V(\boldsymbol{x},\boldsymbol{x}')\Big). \tag{6.6}$$

Thus, the parameter $h \in \mathbb{R}$ from Eq. (6.3) steers the standard deviation of the Gaussian weights w. In practice, the reconstruction u is found by alternating minimisation of u and w.

6.2 Strengths and Weaknesses of the Inpainting Techniques

> *Only strength can cooperate. Weakness can only beg.*
>
> — **Dwight D. Eisenhower**
> (34th President of the United States)

In the following sections we want to assess the advantages and drawbacks of diffusion and exemplar-based inpainting in the context of image compression. Before we consider experiments, let us first briefly recall diffusion-based inpainting, which we have discussed in the previous chapters, and its properties.

The role of the diffusion process is to propagate the known information to the inpainting domain $\Omega \setminus K$. Thereby, the missing parts are filled in. This process of data propagation follows the partial differential equation (PDE)

$$\partial_t u = \text{div}(\boldsymbol{D}\boldsymbol{\nabla} u) \quad \text{on } \Omega \setminus K, \tag{6.7}$$

with reflecting boundary conditions on $\partial\Omega$ and Dirichlet boundary conditions on K. The reconstruction is the non-trivial steady-state for $t \to \infty$. For our purposes we choose the edge-enhancing anisotropic diffusion (EED) [295] from Chapter 3.1, since it has been proven to be particular well-suited for inpainting by Schmaltz et al. [252] and the previous chapters of this thesis. EED uses an anisotropic, structure-adaptive diffusion tensor of the form

$$\boldsymbol{D} := \lambda_1(\boldsymbol{\nabla} u_\sigma)\boldsymbol{v}_1\boldsymbol{v}_1^\top + \lambda_2\boldsymbol{v}_2\boldsymbol{v}_2^\top, \tag{6.8}$$

$$\boldsymbol{v}_1 \parallel \boldsymbol{\nabla} u_\sigma, \quad \lambda_1(\boldsymbol{\nabla} u_\sigma) := g(|\boldsymbol{\nabla} u_\sigma|^2), \tag{6.9}$$

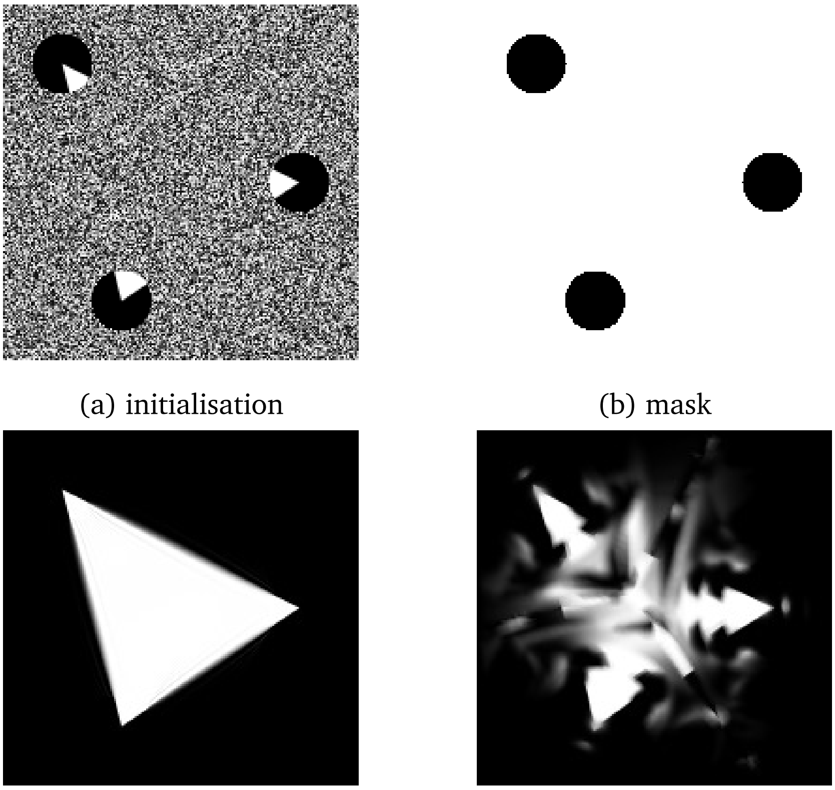
$$\boldsymbol{v}_2 \perp \boldsymbol{\nabla} u_\sigma, \quad \lambda_2 := 1. \tag{6.10}$$

It guides the diffusion process in terms of its eigenvalues λ_1 and λ_2 that specify the amount of diffusion in the direction of the corresponding eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. The tensor design in Eq. (6.8) implies that diffusion *across* edges is inhibited by the Charbonnier diffusivity [54]

$$g(s^2) := \frac{1}{\sqrt{1 + s^2/\lambda^2}} \tag{6.11}$$

with some contrast parameter $\lambda > 0$. Full diffusion *along* edges is achieved with a constant second eigenvalue $\lambda_2 := 1$. Optimising both the positions of the known data, i.e. the inpainting mask K, as well as the contrast parameter λ can improve the reconstruction quality.

Experiments in [252] show that the convolution $u_\sigma := K_\sigma * u$ with a Gaussian K_σ of standard deviation σ in Eq. (6.8) plays an important role for the application of EED in image compression.


(a) initialisation

(b) mask

(c) EED

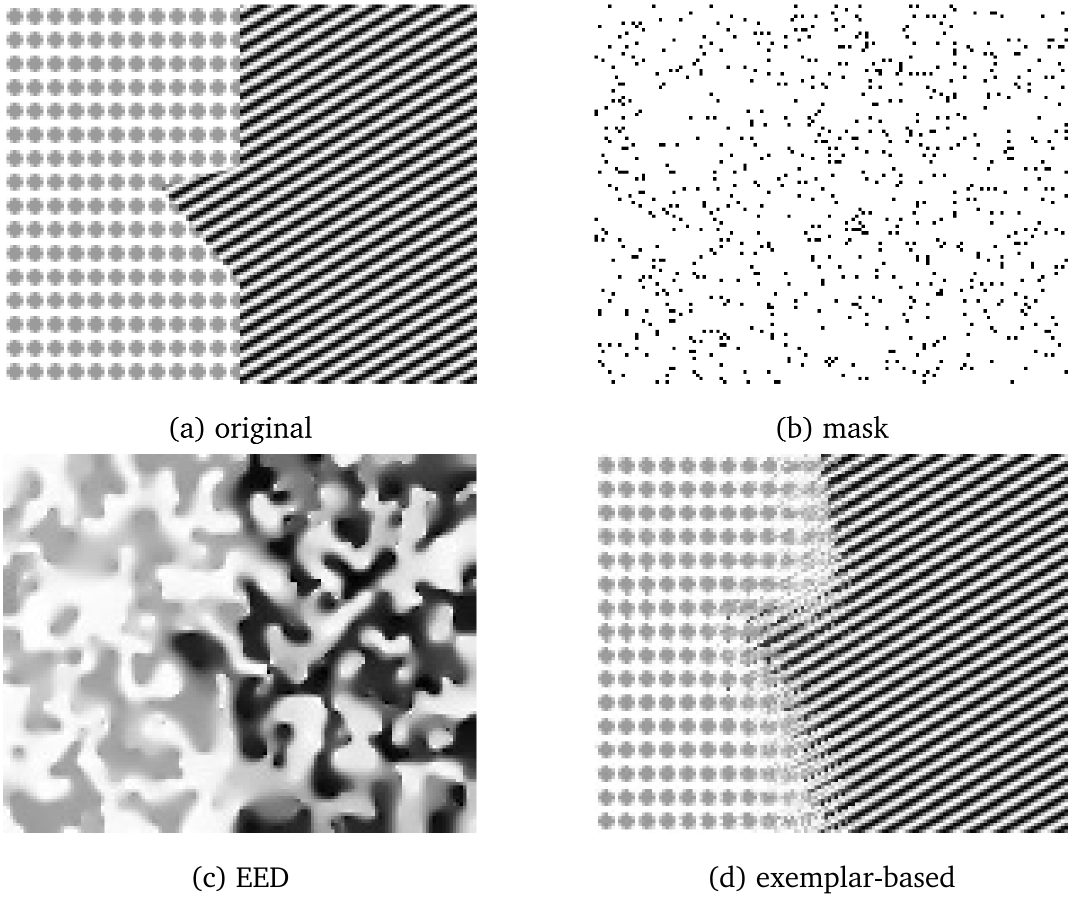
(d) exemplar-based

Figure 6.2.: **Experiment: Structure Propagation** Known data is only given at black locations in the mask. EED ($\lambda = 0.01$, $\sigma = 4$) continues the edge structures into the inpainting domain and reconstructs an almost perfect triangle. The exemplar-based method ($h = 100$) propagates structure only locally and creates copies of structure.

It propagates structural information into the neighbourhood of each pixel and thereby allows reconstruction of edges from a very sparse inpainting mask.

Note that for both diffusion- and exemplar-based inpainting, known data can have a global influence, but the propagation differs significantly nonetheless. In diffusion inpainting, structure is propagated over long distance indirectly through pixel neighbourhoods. Thus, this propagation affects all pixels on the propagation path. Patch-based inpainting, however, is inherently non-local, since each pixel is directly averaged from contributions of similar patches, no matter where they are located. Figure 6.1 illustrates this difference.

Now that we have briefly reviewed both inpainting methods, we demonstrate specific properties with simple synthetic examples and discuss their implications for practical purposes. Let us first consider the capabilities of both algorithms in respect to structure propagation. To this end, we consider a variation of the well-known Kanizsa triangle that was used in [252] to demonstrate the capabilities of EED. For a human observer, the known data in Figure 6.2 (a) suggests that three corners of a triangle are given here, since human perception tends to continue sharp edges.


(a) original

(b) mask

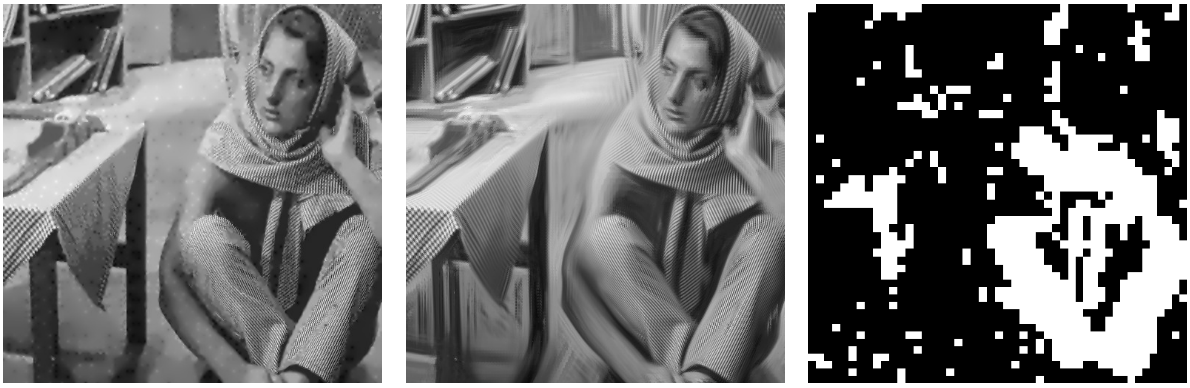
(c) EED

(d) exemplar-based

Figure 6.3.: **Experiment: Texture**. Known data from the original image is only given at black locations in the mask. EED with $\sigma = 0.8$ and $\lambda = 0.9$ completely fails to reconstruct the texture and creates a coarser pattern instead. Exemplar-based inpainting ($h = 150$) reconstructs regular texture very well, but has problems at the interface between textures.

EED is able to preserve sharp edges and propagates image structure due to the locally adaptive diffusion tensor. Therefore, with adequate parameter choices, it is possible to match the expected reconstruction very well (see Figure 6.2 (c)). In contrast, exemplar-based inpainting continues structures only in close vicinity to the known data (Figure 6.2 (d)). In regions of the inpainting domain where known data is far away, structures are copied and multiplied.

For image compression, this behaviour implies that EED is well-suited to reconstruct coarse-scale image features from sparse known data, if they consist of mostly homogeneous areas that are separated by high contrast edges. Exemplar-based inpainting, however, tends to create visually distracting artefacts in such a setting. The reconstructions of the test image *barbara* in Figure 6.4 (a) and (b) illustrate the practical effects well. For example in the face region, exemplar-based inpainting repeats vertical structures of the hood in the cheeks, while EED produces a much more convincing reconstruction. Similar effects can be observed throughout the whole image.



(a) EED

(b) NLI

(c) block decomposition

Figure 6.4.: **Reconstruction of the Image *barbara* with EED (a) and NLI (b) from the Same Known Data.** The reconstructions correspond to intermediate results from steps 1 and 2 of the hybrid algorithm for a compression ratio of $\approx 18 : 1$. The block decomposition (c) with $b = 48$ indicates where EED (black) and NLI (white) yield better results.

In a second synthetic experiment we consider the reconstruction of textured areas. Figure 6.3 displays the test image *interface* from [90], a representative for another extreme type of image content, namely repetitive texture. Here, EED completely fails to reconstruct the texture in a satisfying way. If an isolated region like e.g. a grey dot in the left hand side of the image is not represented by several known pixels that encode both its grey value and its shape, EED has no chance to recreate it. In contrast, exemplar-based inpainting benefits from its tendency to copy structure and create regular patterns. Its reconstruction in Figure 6.3 (d) is fairly close to the original, except for the sharp boundary between both repetitive patterns.

In practical compression applications, EED struggles with repetitive small-scale structure even when a lot of known data is given (e.g. Barbara's trousers in Figure 6.4 (a)). Therefore, compression algorithms that purely use EED for reconstruction have to store textures almost verbatim to achieve a good reconstruction. NLI produces a visually much more pleasing texture inpainting (see Figure 6.4 (b)) that is also close to the ground truth in regard to quantitative error measures. Therefore, a sparse inpainting mask in combination with NLI inpainting can potentially be used for compression.

6.3 A Hybrid Compression Algorithm for Textured Images

> *Combine the extremes,*
> *and you will have the true centre.*
>
> — **Karl W. F. Schlegel**
> (Poet and Philosopher)

Block Decomposition. The core idea of our hybrid algorithm is to combine the strengths of diffusion- and exemplar-based inpainting by decomposing the image into EED and NLI blocks. From a common set of known data, EED blocks are reconstructed with diffusion inpainting and NLI blocks with exemplar-based inpainting. Shared known data for both methods offers two distinct advantages: *storage efficiency* and *direct decomposition*. Since the inpainting mask is only stored once, the only overhead generated by employing two different inpainting methods is the block decomposition and the respective model parameters. In addition, no a priori method for texture/cartoon decomposition is needed. Blocks can be directly classified as EED and NLI blocks by comparing the corresponding reconstructions to the original file, which is (in contrast to the inpainting case) available during compression.

Point Selection and Storage. In the previous sections we have only discussed diffusion and patch-based ideas in an inpainting context, where the inpainting mask $K \subset \Omega$ is already known. For compression, in addition to the right reconstruction method, also a good inpainting mask must be chosen and stored efficiently.

To this end, we employ a rectangular subdivision technique that proves to be successful in the R-EED codec [250, 252]. It limits the choice of the known data positions K to a rectangular adaptive grid that can be represented by a binary decision tree. We iteratively refine this grid by adding known pixels in image regions where the local reconstruction error with EED inpainting exceeds a given threshold. This strategy yields a mask that is optimised for a good diffusion-based reconstruction and efficient storage in form of a binary tree. As soon as the mask is known, we apply a so-called brightness optimisation step: Introducing errors to the small amount of known pixel values can improve the reconstruction quality in the large inpainting domain $\Omega \setminus K$. For more details, we refer to [252]. The optimised grey values are finally quantised and stored with the entropy coder PAQ [178].

Our goal to create a hybrid algorithm requires some modifications to this point selection strategy. Since the subdivision grid adapts to the reconstruction abilities of EED, the point density in an R-EED inpainting mask is low in homogeneous regions, medium near coarse scale edges and very high in textured areas. Since our goal is to reconstruct homogeneous areas and sharp edges with EED and textures with exemplar-based inpainting, this point distribution is not ideal. In particular, textured areas are over-represented in the inpainting mask at the cost of more coarsely quantised grey values. Therefore, we limit the depth N of the binary tree and by that also the minimum grid size of the adaptive inpainting mask.

Avoiding Block Artefacts. For compression algorithms that use block decomposition steps, there is always the danger of visually very distracting discontinuities at block boundaries. In order to keep such effects to a minimum and simultaneously improve the overall reconstruction quality, we propose a modified diffusion-reconstruction in the decompression step. In addition to the known data on K, we also consider the reconstructed NLI blocks as Dirichlet boundary data for the final EED reconstruction. This ensures smoother transitions between NLI and EED blocks and can even improve the EED block reconstructions due to the good approximation of additional known data.

Compression Algorithm. The complete compression pipeline for our hybrid scheme consists of five steps.

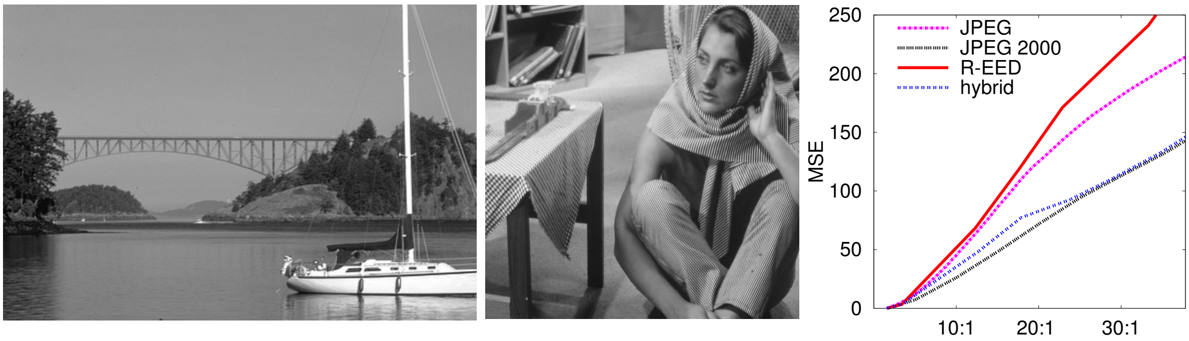
1. **Depth-Limited Subdivision**: Perform rectangular subdivision with a maximum tree depth N to avoid oversampling in highly textured areas. Create a preliminary diffusion reconstruction of the whole image with EED.

2. **Exemplar-Based Inpainting**: Reconstruct the image with NLI and the inpainting mask acquired in the previous step. In order to provide a good prior for structure propagation in non-texture areas to the exemplar-based method, we initialise the inpainting domain $\Omega \setminus K$ with the diffusion reconstruction from Step 1.

3. **Block Decomposition**: Compute a block decomposition: If the mean square error (MSE) of the diffusion reconstruction is lower than the MSE of the exemplar-based inpainting in a given block, consider it to be an EED block, otherwise mark it as an NLI block. Optimising the number b of blocks in x- and y-direction can improve the overall compression quality.

4. **Encoding**: Store the known data in a modified R-EED file format (see [252]) with the number of blocks in the file header. Encode the block decomposition row-wise as a sequence of binary flags for each block (1: EED block, 0: texture block). Here, the context mixing method PAQ [178] yields the best results by encoding the binary tree, block decomposition and grey values jointly.

Decompression Algorithm. Decompression comes down to three straightforward inpainting steps, since the compressed file provides all parameters, known data and the cartoon/texture-decomposition.



(a) *bridge* (b) *barbara* (c) error for *barbara*

Figure 6.5.: **(a)** + **(b)** Original images *bridge* (No. 22090 in the Berkeley database [188]) and *barbara*. **(c)** Error comparison of *barbara* at different compression ratios.

1. **Diffusion-Based Reconstruction**: Extract the inpainting mask and R-EED parameters from the compressed file and reconstruct all missing data on $\Omega \setminus K$ with EED inpainting.

2. **Exemplar-Based Reconstruction**: Initialise $\Omega \setminus K$ with the result from step 1 and perform an NLI reconstruction of the inpainting domain.

3. **Final Inpainting**: Reconstruct the EED blocks with EED inpainting. Use the data of the inpainting mask Ω as well as the reconstructed NLI blocks as known data to improve the final reconstruction.

6.4 Experiments

> *The ultimate court of appeal is observation and experiment, not authority.*
>
> — **Thomas Huxley**
> (Writer)

In the following we evaluate the performance of our hybrid approach in comparison to the R-EED codec and the transform-based coders JPEG and JPEG 2000. For the first step in our hybrid algorithm and the results of R-EED we use the same reference implementation with PAQ for entropy coding. We optimise all model parameters as described in [250]. In our R-EED experiments, we allow larger tree depths N than in the depth-restricted hybrid step wherever this improves the result. For the implementation of NLI in our hybrid scheme we use the publicly available reference implementation of Facciolo et al. [90]. In particular, we apply the fast approximation to the variational NLI scheme AB that is referred to as algorithm O in [90].

The results on the test image *barbara* in Figure 6.5 (c) and Figure 6.6 demonstrate that on images with significant amounts of regular texture content, the hybrid scheme offers a significant quality gain over R-EED. R-EED stores some of the texture almost completely as known data (e.g. in the trouser region) and fails to reconstruct other parts completely (e.g. parts of the hood and tablecloth). Depending on the compression ratio, the hybrid algorithm improves the mean square error (MSE) by more than 50% and is visually much more compelling. It also surpasses JPEG quantitatively by a large margin and does not suffer from similarly obvious artefacts. For compression ratios larger than 25:1, the hybrid algorithm is quantitatively on par with JPEG 2000. While it is not able to beat JPEG 2000, yet, it is the first time that a diffusion-based algorithm achieves comparable results on images with such a high amount of texture.

On images with irregular texture, e.g. *bridge* from Figure 6.5 (a), the quality gain of the hybrid algorithm over R-EED is less significant, but can still reach around 10% depending on the compression ratio. While the image quality is quantitatively worse than JPEG, it is subjectively better due to the absence of block artefacts, especially in zoom-ins.

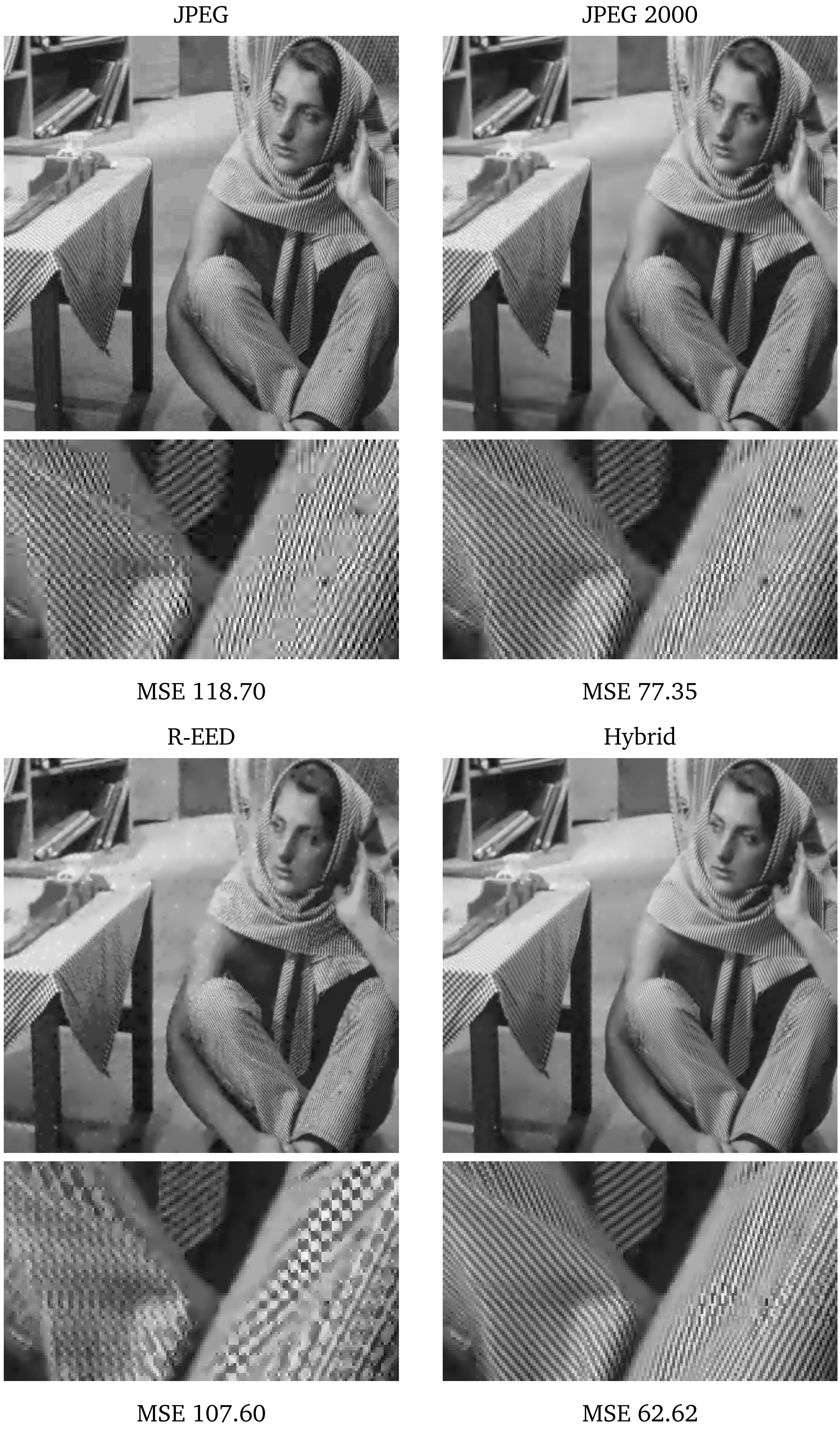


Figure 6.6.: **Results for *barbara* with Ratio $\approx 18:1$.** The hybrid algorithm uses the upper tree limit $N = 16$, block parameter $b = 48$, and the NLI parameter $h = 100$.

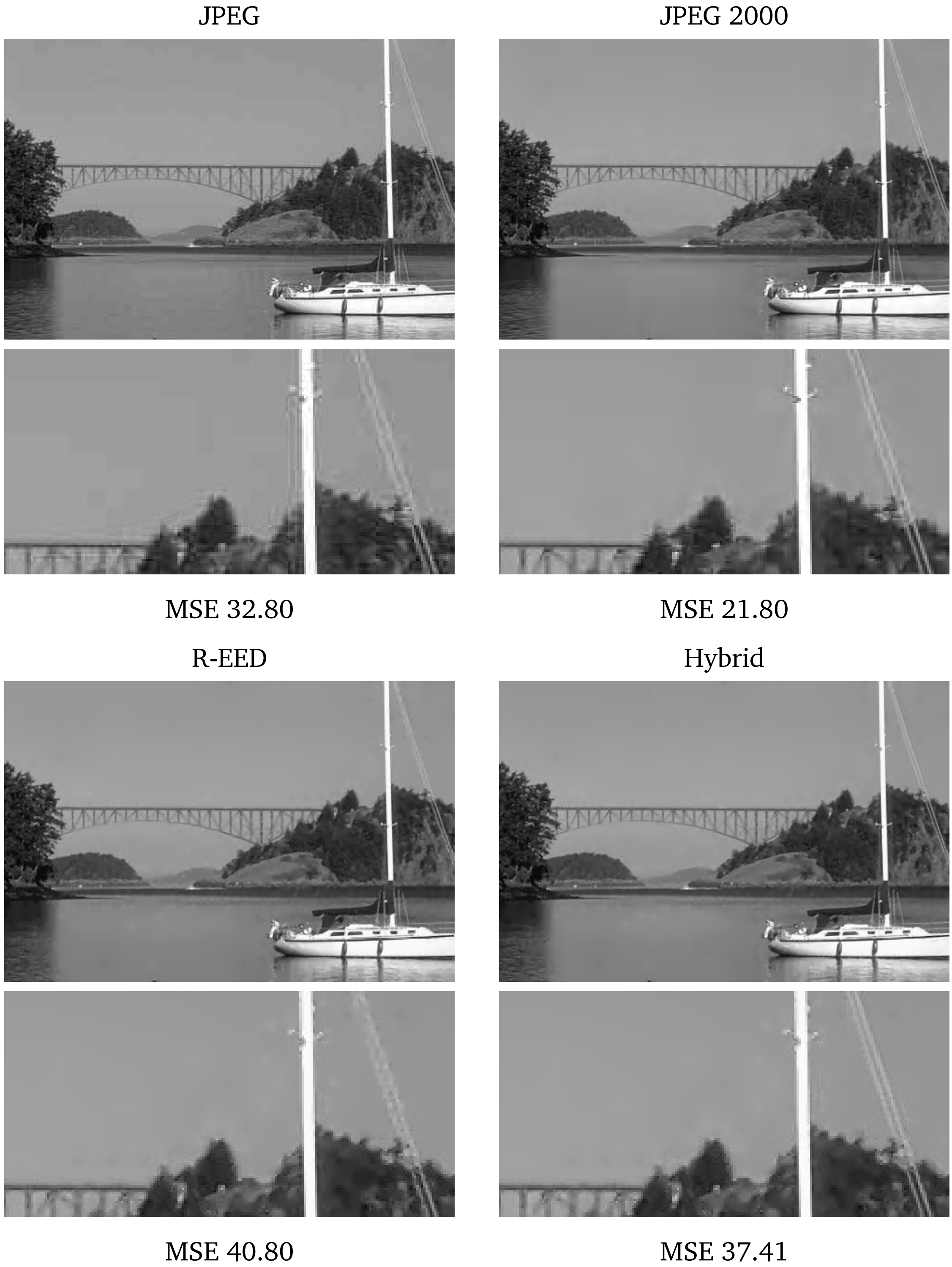


Figure 6.7.: **Results for *bridge* with Ratio** $\approx 19 : 1$**.** The hybrid algorithm uses the upper tree limit $N = 16$, the block size $b = 62$, and the NLI parameter $h = 25$.

6.5 Conclusion

> *It's more fun to arrive at a conclusion than to justify it.*
>
> — **Malcolm Forbes**
> (Publisher)

With the first combination of diffusion-based and exemplar-based inpainting from sparse data, textured images can be compressed efficiently. Our hybrid algorithm uses the full spectrum of inpainting ideas for compression while keeping the resulting overhead small. This is an important step towards closing the gap between the widely accepted general purpose encoders JPEG and JPEG2000 and diffusion-based methods that have thus far shown their advantages primarily in more specialised applications like depth-map or cartoon encoding.

For future work it would be particularly interesting to investigate how far the quality could be improved further, if exemplar-based inpainting is treated equally with EED instead of using it as a postprocessing step. In particular, a lot of additional potential lies in optimising the choice of known data for a good trade-off between quality in EED and NLI reconstructions.

It should also be noted that the block-decomposition proposed in this chapter is not limited to a combination of EED and the NLI approach of Facciolo et al. [90]. Both methods can be exchanged, if more efficient alternatives should arise in the future. Moreover, we use the subdivision strategy of R-EED to find known data. This means that with minor modifications, our framework is compatible with the region of interest coding from Chapter 8 or the progressive modes of Schmaltz et al. [251]. Thus, it is flexible enough to act as a foundation for potential future hybrid algorithms.

Publication Info

The contents of this chapter were introduced by Peter and Weickert [220] on the International Conference on Scale Space and Variational Methods in Computer Vision 2015. This thesis goes into more details at some points and provides additional visualisations, but the algorithm and results have not been modified compared to the original publication.

7 Immediacy: Real-Time Video Decoding

> *Time is the most valuable thing a man can spend.*
>
> — **Teophrastus**
> (Philosopher)

The first chapters of this thesis, as well as most previous publications [99, 100, 250, 252] on PDE-based compression, focused purely on reconstructing the original image as accurately as possible at a given compression ratio. The *immediacy* aspect, providing reconstructions to the user in a short amount of time, was not considered at all. However, in time critical applications such as video playback, the runtime of the algorithm is essential.

In a first step towards real-time applications, we want to be able to compress full-length movies in a feasible amount of time and be able to play them back in real-time. At this point, competing qualitatively with the established video coders is not feasible, yet. In contrast to image compression, real-time decompression is a minimum requirement for video data and has therefore to be addressed before quality. Consequently, we aim at a proof-of-concept solution that preserves the compression quality of existing PDE-based coders without exploiting additional redundancies in the time-domain of the video compression setting.

Our Contribution. We present a new PDE-based video compression codec based on R-EED that is specifically designed for real-time playback and can act as a basis for future codecs. It organises the video into groups of pictures (GOPs) that allow to adapt compression parameters locally to the scene content. Furthermore, these GOPs provide the necessary structure for caching and random access in video playback. To achieve real-time performance of the decoder, we examine the performance of R-EED with different numerical solvers and implement a player that relies on parallelisation on the GPU and efficient reuse of already inpainted frames. We evaluate our algorithm on a realistic real-world example for monochrome video data: The famous vampire movie *Nosferatu* by F. W. Murnau [87].

Related Work. Almost the whole existing body of work on PDE-based compression focuses on still image compression. However, there are some notable exceptions from the rule. So far, the only codec that is fully PDE-based and qualitatively competitive to transform-based approaches is the model-based approach by Schmaltz and Weickert [249]. It combines PDE-based image compression for static backgrounds with tracking of moving objects. These moving objects can then be replaced by 3-D models and only the movement has to be encoded. However, this method is only applicable to a narrow field of videos in practice (e.g. video conferencing).

Gao [101] used diffusion-based inpainting to compress optic flow fields for motion compensation in video coding. However, the actual compression of video data in this approach relies on block-coding with the discrete cosine transform. Therefore, the codec is still mainly transform-based. In a similar fashion, Doshkov et al. [77] also just replace one step of a transform-based coder. They use linear homogeneous diffusion for prediction: Already encoded data is prescribed as boundary conditions and the diffusion inpainting provides a prediction of the remaining unknown values.

So far, real-time video decoding with fully PDE-based general purpose algorithms has only been implemented by Köstler et al. [156] and Baum [22]. The work of Köstler et al. considers 320×200 videos on a Playstation 3 while the approach of Baum deals with a resolution of 640×480 like our own experiments. It only achieves real-time decompression at the price of a noticeably degraded compression quality.

Organisation of the Chapter. Before we define our actual codec, we first discuss how the challenges of real-time performance can be overcome by a combination of efficient numerical solvers (Section 7.1) and reuse of already decompressed frames (Section 7.2). Both techniques are prerequisites for the real-time capabilities of the proof-of-concept codec that we propose in Section 7.3. The architecture of the corresponding video player is discussed in Section 7.4. In Section 7.5, we compare the results of our codec to the ground truth obtained from a decompression without time constraints. Finally, we conclude the chapter with a short summary in Section 7.6.

7.1 Numerical Solvers

> ” *A problem well put is half solved.*
>
> — **John Dewey**
> (Writer)

Our goal is to compress a full movie frame-by-frame with the R-EED algorithm from Chapter 3.3 and decompress it in real-time. Therefore, the most important component for fast compression and decompression is an efficient implementation of inpainting with EED. In R-EED, the inpainting problem is described by the elliptic PDE

$$(1 - c_K)\,\mathrm{div}(\boldsymbol{D}(\boldsymbol{\nabla} u_\sigma)\boldsymbol{\nabla} u) - c_K\,(u - f) = 0, \tag{7.1}$$

a special case of the general inpainting equation (3.25) from Chapter 3.1. There are different approaches to discretise and solve this continuous inpainting problem. In the following we discuss solvers from previous implementations of R-EED as well as recent cyclic schemes and compare their suitability for real-time decompression.

One possibility to acquire the inpainted image is to compute the steady state of the evolution

$$\partial_t u = \text{div}(\boldsymbol{D}\boldsymbol{\nabla} u) \quad \text{on} \quad \Omega \setminus K \times (0, \infty), \tag{7.2}$$

$$u(\boldsymbol{x}, t) = f(\boldsymbol{x}) \quad \text{on} \quad \partial K \times (0, \infty), \tag{7.3}$$

$$\langle \boldsymbol{D}\boldsymbol{\nabla} u, \boldsymbol{n} \rangle = 0 \quad \text{on} \quad \partial \Omega \times (0, \infty). \tag{7.4}$$

In the previous chapters, we have mainly relied on this parabolic formulation of the inpainting problem. A straightforward and easy to implement approach is the explicit discretisation of the parabolic formulation (7.2). An explicit scheme discretises both the temporal and spatial derivatives with suitable finite difference approximations (for more details see [299]). This yields an iterative scheme of the form

$$\frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^k}{\tau} = \boldsymbol{A}(\boldsymbol{u}^k)\boldsymbol{u}^k \tag{7.5}$$

$$\Leftrightarrow \quad \boldsymbol{u}^{k+1} = (\boldsymbol{I} + \tau \boldsymbol{A}(\boldsymbol{u}^k))\boldsymbol{u}^k. \tag{7.6}$$

Here, $\boldsymbol{A}(\boldsymbol{u}^k)$ is the discretisation of the spatial derivative operators from Equation (7.2), and τ is the discrete time step size. For stability reasons, the time step size is severely limited and thus the scheme requires a large number of iterations to reach the steady state. However, fast explicit diffusion (FED) [114], a recent cyclic scheme, makes efficient implementations of explicit approaches possible. In particular, FED is well-suited for parallelisation and performs exceptionally well on graphics processing units.

Alternatively, one can also solve the elliptic formulation (7.1) of the inpainting problem directly. Using the same spatial discretisation $\boldsymbol{A}(\boldsymbol{u})$ of the divergence term as before, one obtains

$$(\boldsymbol{I} - \boldsymbol{C}_K)\boldsymbol{A}(\boldsymbol{u})\boldsymbol{u} - \boldsymbol{C}_K(\boldsymbol{u} - \boldsymbol{f}) = 0. \tag{7.7}$$

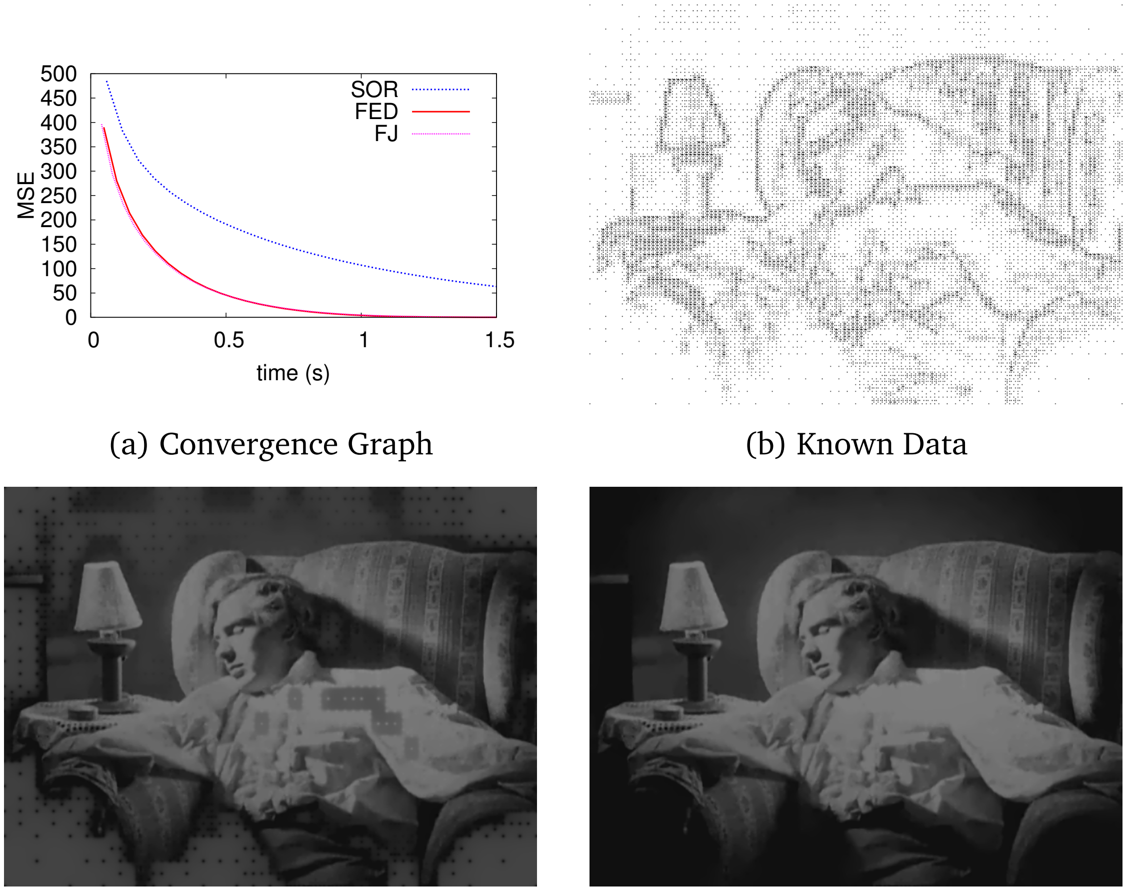
Here, $\boldsymbol{C}_K$ is a quadratic diagonal matrix that contains the entries of the discrete inpainting mask vector $\boldsymbol{c}_K$. Following the approach of Mainberger et al. [184], Equation (7.7) can be rearranged into the nonlinear system of equations

$$\underbrace{-((\boldsymbol{I} - \boldsymbol{C}_K)\boldsymbol{A}(\boldsymbol{u}) - \boldsymbol{C}_K)}_{=:\boldsymbol{M}(\boldsymbol{u})} \boldsymbol{u} = \boldsymbol{C}_K \boldsymbol{f}. \tag{7.8}$$

We obtain the solution of this system by the fixed point iteration

$$\boldsymbol{M}(\boldsymbol{u}^k)\boldsymbol{u}^{k+1} = \boldsymbol{C}_K \boldsymbol{f}. \tag{7.9}$$

For each iteration k, a linear system of equations has to be solved. Previous publications on PDE-based compression [250, 252] use successive overrelaxation (SOR) (see e.g. [287]) for this task. Unlike FED, SOR is in essence a sequential algorithm. Therefore, there is virtually no potential for increasing its performance by parallelisation.



(a) Convergence Graph

(b) Known Data

(c) FED Inpainting after 0.04s

(d) Reference Inpainting

Figure 7.1.: **Comparison of Numerical Solvers. (a)** Mean square error in respect to reference solution over time on frame 130254 of the movie *Nosferatu* (licensed by the F.W. Murnau foundation). **(b)** Known data selected by the R-EED subdivision scheme. Pixels from the set of mask locations K are marked in black, unknown pixels in white. **(c)** EED-inpainting results after ≈ 0.04 seconds for FED with flat initialisation with the average grey value of the known data. Homogeneous regions with a low density of known data are not fully inpainted. **(d)** Reference inpainting.

However, there is also a cyclic extension of the Jacobi overrelaxation scheme (see e.g. [287]) by Weickert et al. [298], the Fast-Jacobi (FJ) method. Since the cyclic FJ solver can be more efficient than its parabolic counterpart FED, in particular for elliptic problems with strongly varying coefficients, we also consider it for our purposes.

In order to analyse the suitability of solvers for real-time video decoding with R-EED, we compare FED, FJ, and SOR with respect to their convergence behaviour. To this end, we compute a reference solution for R-EED compressed frames of the movie *Nosferatu* [87] by iterating the explicit scheme until convergence. Each frame has a resolution of 640×480 and we initialise the missing image parts with the average grey value of the known data. For all three solvers, Fig. 7.1 (a) provides the evolution over time of the mean square error (MSE) in relation to the reference inpainting. It becomes clear immediately that FJ does not offer a real advantage compared to FED in this context, their convergence behaviour is almost identical.

On a test system with an *Intel Xeon CPU W3565@3.20GHz* and an *Nvidia Geforce GTX 460*, a parallelised explicit scheme on the GPU typically takes almost 5 minutes to converge. SOR requires around 6 seconds and parallelised FED still needs approximately 2 seconds for convergence. For ill-posed frames, the total inpainting runtime can be even longer.

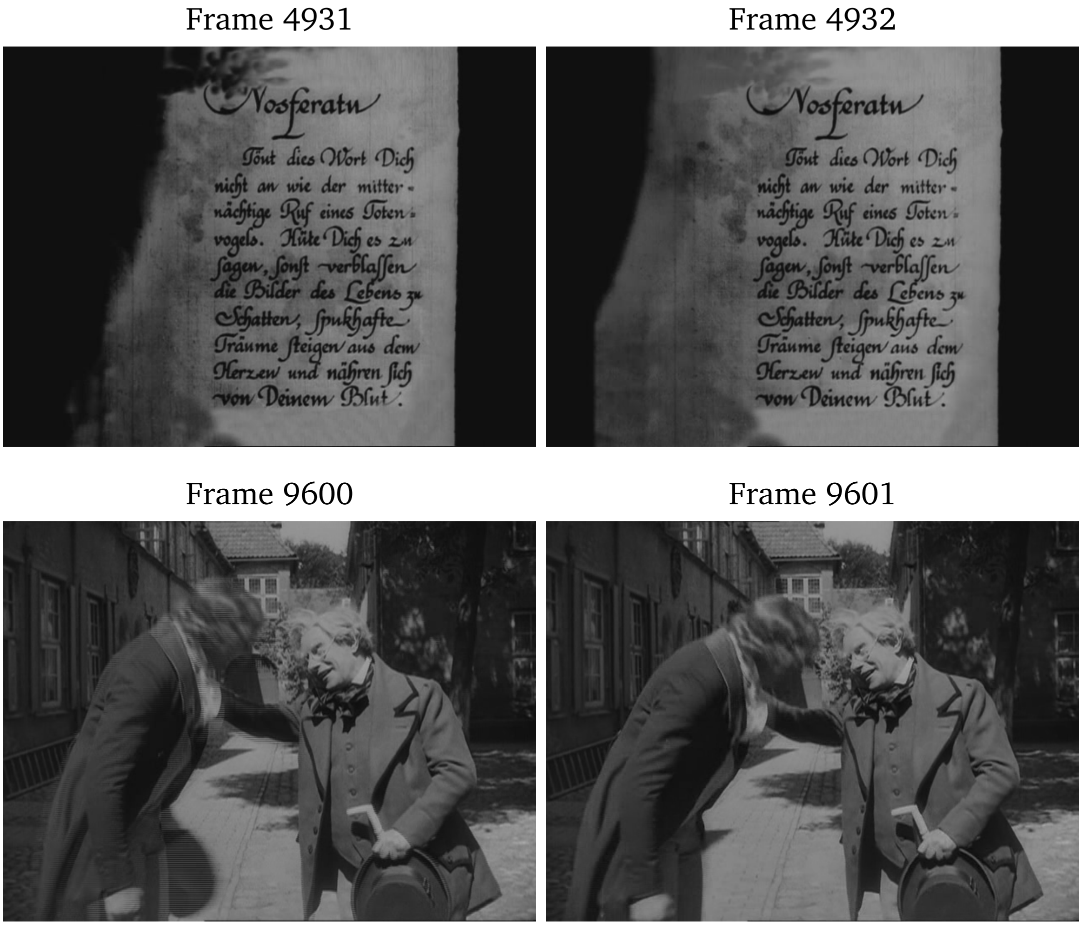
The comparison in Fig. 7.1 clearly demonstrates that even the advanced cyclic algorithms with parallelisation cannot achieve real-time video decoding on consumer hardware. Fig. 7.1 (b) and (c) show the known data and a reconstruction with FED after a runtime of 0.04 seconds. This is the time budget that is available to achieve 25 frames per second. Clearly visible artefacts that are not present in the converged reference inpainting (Fig. 7.1 (d)) make these results unsuitable for video playback.

7.2 Smart Initialisation

> *The purpose - where I start - is the idea of use.*
> *It is not recycling, it's reuse.*
>
> — **Issey Miyake**
> (Designer)

One way to reduce the runtime of iterative diffusion schemes is to provide a good initalisation. So-called coarse-to-fine approaches [29] already successfully apply this idea. First, they generate a pyramid of subsampled versions of the inpainting mask. On the coarsest level, i.e. the smallest representation of the image, inpainting is performed. The upsampled result of this inpainting provides an initialisation for the next finer level of the pyramid. A successive ascend through the finer levels of the pyramid finally yields a good initialisation for an inpainting of the full image. Such approaches can be interpreted as special cases of multigrid solvers [37, 117].

While coarse-to-fine approaches yield a substantial increase in performance, subsampling and diffusion on coarser scales creates some overhead. Furthermore, the benefits of parallelisation are less pronounced on coarse levels due to the smaller number of possible simultaneous calculations. In video sequences, however, there is an additional way to obtain suitable initialisations: In many cases, subsequent frames are very similar. Therefore, the inpainting result of an already processed frame can be reused as an initialisation for the next frame. For groups of pictures (GOPs) that are sufficiently similar, the reuse of inpainting results provides a significant speed-up without any additional computational cost. Consequentially, the core idea


Frame 4931

Frame 4932

Nosferatu

Tönt dies Wort Dich nicht an wie der mitter-nächtige Ruf eines Toten-vogels. Hüte Dich es zu ſagen, ſonſt verblaſſen die Bilder des Lebens zu Schatten, ſpukhafte Träume ſteigen aus dem Herzen und nähren ſich von Deinem Blut.

Frame 9600

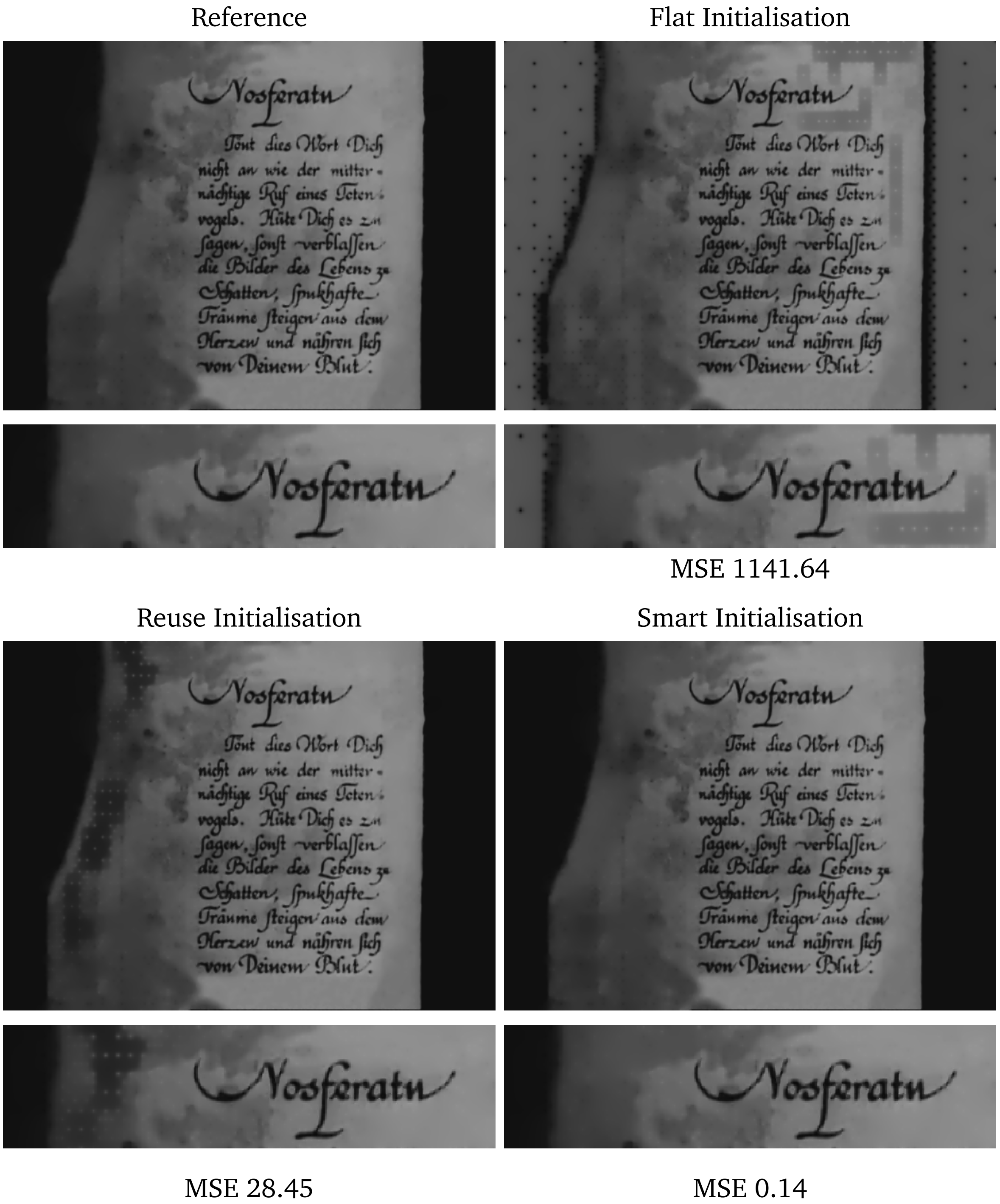
Frame 9601

Figure 7.2.: **Scenes with Quick Movement.** In frames frames 4932 and 9601 and and their predecessors, quick movement takes place. From frame 4931 to 4932 a book page is flipped. Frames 9600 and 9601 contain quick movement of a hat in the hand of the character to the left.

of the smart initialisation scheme is to segment a video into such GOPs that allow to reuse as much already computed information as possible.

In some cases, the initialisation with the previous frame is not useful or more difficult, though. In particular at scene transitions or in case of rapid camera movements, the content of subsequent frames can change drastically and make this initialisation infeasible. Motion can also be a problem, if it affects only small areas of the video. It is possible that the initialisation is very good for large parts of the frame, but unsuitable in the vicinity of moving objects. For example, a character rapidly moves a dark object in front of a bright background in the frames 9600 and 9601 of the movie *Nosferatu* (see Fig. 7.2). This generates almost a worst case initialisation.

However, frames that are partially unsuitable for reuse can still be useful. Initialisations that deviate significantly from the final steady state are only problematic if they occur in areas with a low density of known points. If many known data is given in the corresponding region, their information does not need to be propagated over large distances. Thus, a small number of



Reference

Flat Initialisation

MSE 1141.64

Reuse Initialisation

Smart Initialisation

MSE 28.45

MSE 0.14

Figure 7.3.: **Reconstruction of Frame 4932 based on Different Initialisations (Compression Ratio 20:1).** The top row displays the reference inpainting and a reconstruction that uses a flat initialisation with the average grey value of the known pixels. In the second row, the last frame was used as initialisation on the left, and on the right linear interpolation is used in addition based on a confidence measure. Smart initialisation provides the lowest MSE for the total reconstruction.

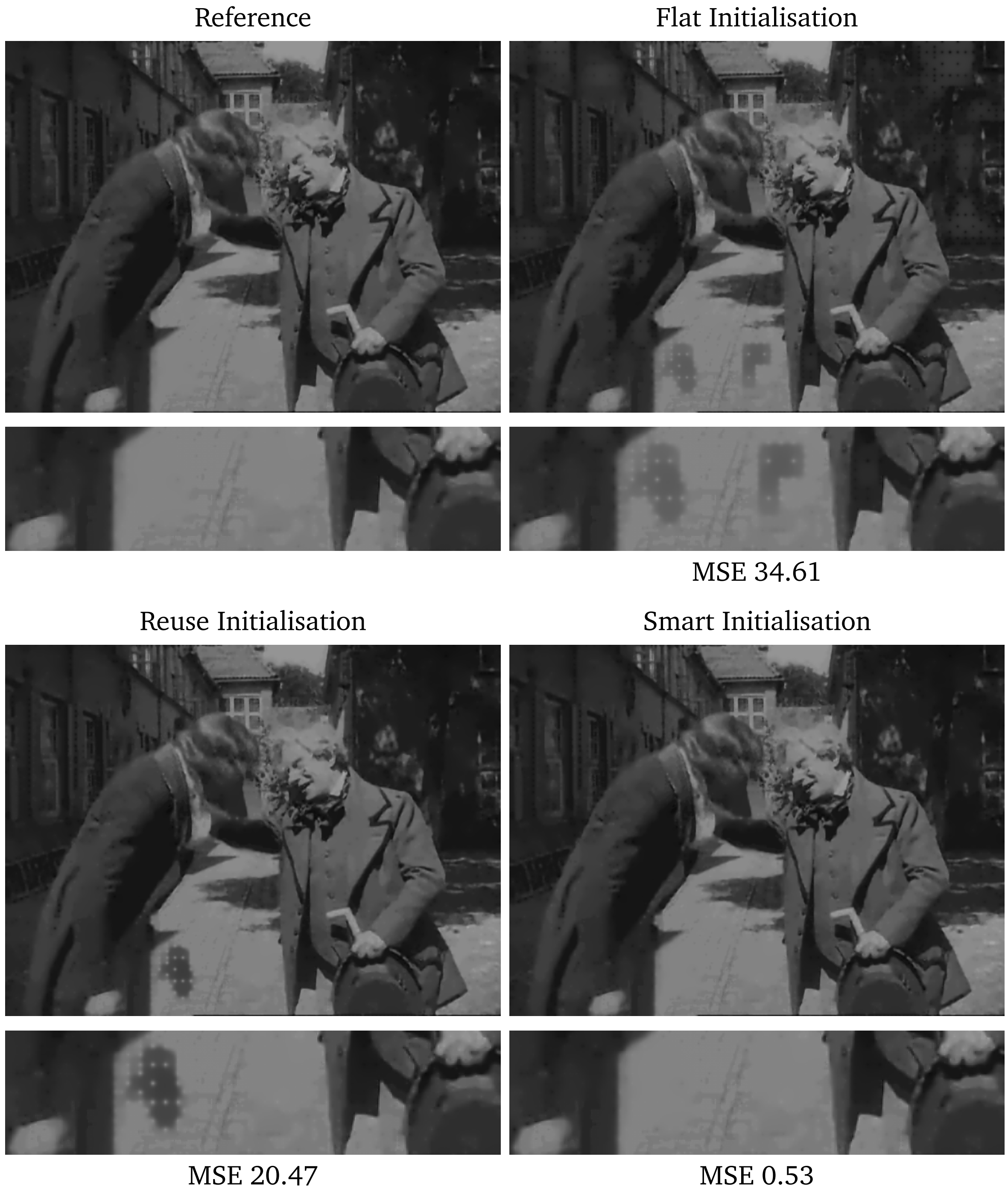


Figure 7.4.: **Reconstruction of Frame 9601 based on Different Initialisations (Compression Ratio 20:1).** The top row displays the reference inpainting and a reconstruction that uses a flat initialisation with the average grey value of the known pixels. In the second row, the last frame was used as initialisation on the left, and on the right linear interpolation is used in addition based on a confidence measure. Smart initialisation provides the lowest MSE for the total reconstruction.

iterations is sufficient to reach a good approximation to the steady state. Since diffusion only needs a very low amount of known data to reproduce homogeneous image regions, these are prone to initialisation problems. However, in flat image areas, a good approximation to the steady state can be achieved by simple and fast linear interpolation.

Therefore, we use a confidence function to decide if parts of the initialisation with the last frame should be replaced by linear interpolation results. Let a rectangular image section from the R-EED subdivision be given by the set P of its corner points:

$$P := \{(x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2)\}. \tag{7.10}$$

The confidence in the linear interpolation quality in this rectangle is influenced by two factors: the size of the subimage and its homogeneity. We define the size factor s_P as the maximum side length of the rectangular region:

$$s_P := \max\{|x_2 - x_1|, |y_2 - y_1|\}. \tag{7.11}$$

The larger the region, the larger the expected error of the linear interpolation with the known points from this rectangle. However, this error also depends on the homogeneity of the region. If the rectangular region is perfectly flat, the linear interpolation will be flawless, no matter the size of the region. The more discontinuities appear in the rectangle, the less accurate it becomes.

Therefore, we also incorporate a homogeneity factor h_p. Since only the known image points can be used to determine this aspect of the region, we define h_P as

$$h_P := \max_{(x,y)\in P} f(x, y) - \min_{(x,y)\in P} f(x, y). \tag{7.12}$$

For large maximal deviations of the known grey values in the region, less accurate results can be expected from the linear interpolation.

A multiplicative combination of the size and homogeneity factors yields the overall confidence function c_P:

$$c_P := \psi(s_P) \cdot h_P. \tag{7.13}$$

Here ψ is a truncated quadratic function with a threshold parameter ξ (default value $\xi = 30$) that also rescales the range of the size factor to the interval $[0, 1]$:

$$\psi(x) = \min(x^2, \xi)/\xi. \tag{7.14}$$

The threshold ξ and the rescaling avoids that the size factor dominates the confidence measure. This can happen since the homogeneity factor is limited to the range $[0, 255]$ while the image size is not limited. For image sections with a height or width smaller than ξ, the homogeneity factor is attenuated. For larger sections, it has the full impact.

The two examples in Figure 7.3 and Figure 7.4 illustrate that the use of the confidence-dependent linear initialisation in smart initialisation offers significant improvements over a pure reuse initialisation. Experimental results on a large number of images are discussed in Section 7.5.

7.3 An EED-based Framework for Video Playback

> *A scene has to have a rhythm of its own,*
> *a structure of its own.*
>
> — **Michelangelo Antonioni**
> (Director)

In the following, we use the fast inpainting techniques from the previous section to build a framework for real-time video reconstruction with anisotropic diffusion inpainting. Note that we solely focus on the aspect of real-time reconstruction and not on qualitative competitiveness to transform-based coders. For this proof-of-concept codec, we want to preserve the quality of PDE-based still-image compression. To this end, we first present a wrapper that extends R-EED to video sequences and the necessary structures for frame reuse. In future publications, the R-EED wrapper can act as a starting point for true PDE-based video encoders that explicitly exploit temporal redundancies in the video. The second part of the video player framework is the real-time decoder that we discuss in detail in Section 7.4.

The core structure of our framework relies on a segmentation of the video in groups of pictures (GOPs). The primary purpose of these GOPs is to define regions where smart initialisation is feasible. Therefore, we employ scene change detection to create a temporal segmentation.

Smart initialisation always requires the reconstruction of the previous frame to decode the current frame. Therefore, the frames in a GOP require successive decoding starting with the first frame of the GOP. This implies that random access in the video (e.g. for fast forwarding or chapter selection) is limited to the beginning of GOPs, similar to common video compression codecs such as the ones from the MPEG and H26X family. Due to this constraint, we also limit the maximal number of frames N in a GOP. For practical purposes, we choose $N = 32$ in this thesis.

Each GOP is essentially a collection of up to N R-EED encoded images. Since by definition the frames in a GOP have similar content, we can choose a common R-EED contrast parameter λ and a common quantisation parameter q for the whole GOP. All other R-EED parameters are identical for the whole video and thus only the inpainting mask has to be stored for individual frames. For sufficiently similar frames, it is possible to reuse the mask positions or even the whole inpainting mask including the grey values without a significant loss in compression quality. Therefore, we introduce four different frame types which we categorise by the similarity to previous frames. Depending on the frame type, the framework stores different parts of the inpainting mask.

- **GOP start (type 1):** Always contains the binary tree that encodes the known point positions and the grey values of the inpainting mask. This guarantees that all information to reconstruct this frame independently from other frames is available. The framework assigns type 1 to a frame, if the maximum number N was reached in the previous GOP or the difference to the previous frame exceeds a threshold $t_1 > 0$ (default value: $t_1 = 15$). Additionally, we create a new GOP, if the difference between the reference steady state and the reconstruction exceeds a threshold $t_{\text{ref}} > 0$ (default value: $t_{\text{ref}} = 5$). This additional constraint avoids error spikes at ill-posed images, e.g. those that contain interlacing artefacts.

- **No mask reuse (type 2):** For this type, the framework does not reuse point positions, just the initialisation and R-EED parameters. Just like for a type 1 frame, we store both the binary tree and the quantised grey values. The framework assigns type 2, if the difference to the previous frame exceeds the threshold $t_2 \leq t_1$ (default value: $t_2 = 5$).

- **Tree reuse (type 3):** The point positions from the previous frame remain unchanged, but we store different quantised grey values for these positions. The framework assigns type 3, if the difference to the previous frame exceeds the threshold $t_3 \leq t_2$ (default value: $t_3 = 1$).

- **Full mask reuse (type 4):** We consider this frame to be fully redundant and no additional data is stored. The framework assigns type 4, if the difference to the previous frame does not exceed the threshold t_3.

Just as in the regular R-EED scheme, we store the mask positions as a binary sequence of splitting decisions and a minimum and maximum tree depth. An entropy coder saves the grey values efficiently. Furthermore, a GOP header contains all information that is needed to decode the GOP frames. It provides the content-dependent R-EED parameters λ and q, as well as a list of frame types. Additionally, the header includes a list with the lengths of each GOP data set to enable separation of the frame data for all possible entropy coders.

Finally, all of the GOP data must be included in a single video file. This container format consists of a global video header and a concatenation of all GOP data. The global header contains all parameters that do not have to be adapted to the specific image content: video resolution, total number of frames, a colour flag, the entropy encoding type, the point pattern for rectangular subdivision (see [252]), and the progressive mode parameters. In order to allow random access to the GOP start frames, the header also includes the length of all GOPs. This enables the video player to jump to arbitrary GOP data.

For the global header and all GOP headers, the parameters are encoded directly as binary numbers of custom length. The frame and GOP lengths are stored efficiently with Golomb coding [106].

7.4 Architecture of the Real-Time Decoder

> *Multitasking? I can't even do two things at once.*
> *I can't even do one thing at once.*
>
> — **Helena Bonham Carter**
> (Actress)

Since the main goal of our framework is real-time video playback, the decoder has to provide reconstructions of a rate of 25 frames per second. This implies that each reconstruction must be available in a time window of 0.04s. Consumer class GPUs can perform around 100 iterations with 5 linear updates for FED solvers in this time.

Additionally, there is some overhead due to file operations, entropy decoding, smart initialisation, and data transfer between CPU and GPU. Moreover, at GOP start frames, there is no initialisation with the reconstruction from the last frame available. In order to minimise overhead and handle start frames, we propose a multi-threaded decoder with caching that makes use of widely available multicore CPUs.

The decoder relies on four distinct threads that operate in parallel and only communicate over a system of caches for initialisation and reconstructed image data.

- **Loader:** The loader parses header data and extracts compressed data for a single frame from the video file. It rebuilds the inpainting mask from stored tree data and decompresses the entropy coded grey-value data. Finally, it computes the linear interpolation for smart initialisation. In order to use times of low CPU usage efficiently, a *GOP init cache* collects the complete inpainting mask and computed linear initialisation for multiple GOP start frames ahead of time. Similarly, the *frame init cache* provides the initialisation data for all other frames. These caches are separated due to their different amount of localisation. The initialisations for GOP start frames correspond to video parts that have a larger temporal distance to the currently displayed frames in order to enable random access and avoid quality loss through inpainting activity spikes in video regions with rapid scene changes. In contrast, precomputed initialisations for other frames only cover GOPs in a close vicinity to the current frame.

- **GOP Starter:** Another thread is entirely dedicated to generating an initialisation for GOP-start frames with coarse-to-fine inpainting on the CPU. It reads the mask and known grey values from the GOP init cache, reconstructs the start frames and stores them in a third cache, the *GOP start cache*. In order to avoid playback pauses, slowdowns or dropped frames, the start handler dynamically adapts the number of iterations and nonlinear updates of the numerical solver to the number of cached frames. If the cache is in danger of running empty, the number of iterations is reduced, if it is sufficiently full, the number of iterations is increased. Thereby, fluent playback is maintained at the cost of a slight degradation in reconstruction quality for the GOP start frames. This is e.g. relevant for user interaction with random access or temporary load spikes of the CPU due to other applications or hard-to-compress sections in the video.

- **Reconstructor:** The reconstructor is responsible for the final inpainting of the frames. Its main task is to transfer initialisation data to the GPU, perform FED inpainting and store the finished reconstruction in the *frame cache*. For all frames except GOP start frames, it also generates the final initialisation. According to the confidence measure for the smart initialisation, it combines the linear interpolation data from the frame init cache with the last reconstruction from the frame cache. Just as the GOP starter, the reconstructor also adapts the number of iterations and nonlinear updates to the number of cached frames to guarantee fluent playback (if the cache is nearly empty) or increase the fidelity (if the cache is sufficiently full).

- **Presenter:** The presenter handles all user interaction and video-playback with the OpenGL toolkit *GLUT*. It reads the decompressed images from the frame cache at a rate of 25 frames per second.

The architecture of the decoder allows to dedicate more time to the actual inpainting by focussing the time-critical GPU interpolation to a single thread and distributing all other tasks to the three other threads. Additionally, the caching system even allows to increase the inpainting runtime above 0.04s, if a larger time budget is available. This can happen if time is saved due to the reuse of whole frames (frame type 4) or if the user pauses the video. Figure 7.5 provides a visualisation of a typical interaction between the different threads in the real-time decoder.

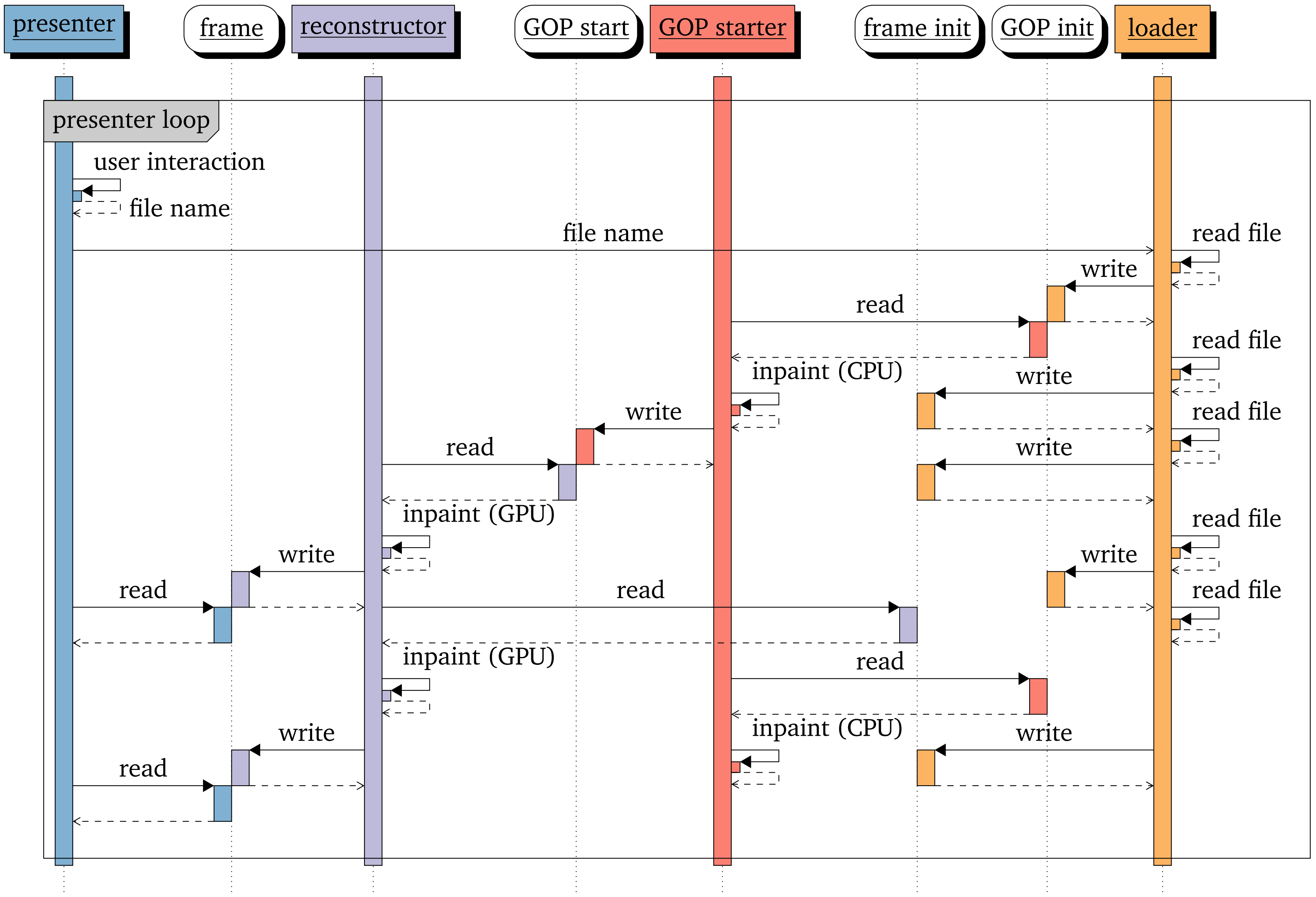


Figure 7.5.: Thread Interaction in Video Decompression. In the example above, the following happens: 1. The loader reads a full GOP with three frames and the first two frames of the next GOP. 2. The GOP starter generates two initialisations for GOP start frames. 3. The reconstructor inpaints one GOP start frame and one regular frame. 4. The presenter displays the two fully inpainted frames.

7.5 Experiments

> *I'm one of those people who says, "yes, cinema died when they invented sound."*
>
> — **George Lucas**
> (Director)

In the following we present experiments that assess the quality of the decoded images with different initialisation strategies. Since the decoded results should approximate the steady-state as accurately as possible, we compare them against reference results.

These reference inpaintings result from an explicit scheme with FED acceleration. As a stopping criterion, we used the maximal per-pixel change after a stopping time of $T = 10000$ with 20 nonlinear updates. For the initialisation $\boldsymbol{u}^k$ and the corresponding reconstruction $\boldsymbol{u}^{k+1}$ after a diffusion time of 10000 we define the per-pixel change

$$d(\boldsymbol{u}^k, \boldsymbol{u}^{k+1}) := \max_{i,j}\{|u_{i,j}^k - u_{i,j}^{k+1}|\}. \tag{7.15}$$

For $\boldsymbol{u}^0$ we use a flat initialisation with the average grey value of the known data. The FED inpaintings with stopping time $T = 10000$ are then iterated until $d(\boldsymbol{u}^k, \boldsymbol{u}^{k+1}) < 0.001$.

All tests were performed on an *Intel Xeon CPU W3565@3.20GHz* with an *Nvidia Geforce GTX 460*. A restored version of the classical movie *Nosferatu* [87] from 1922 acts as a test file. The frames in Figure 7.2, Figure 7.3, Figure 7.4, and (see Figure 7.1) are copyrighted by the Friedrich-Wilhelm-Murnau-Foundation and are published with their consent. It has 141680 grey scale frames at a resolution of 640×480. For our testing purposes, the full movie was encoded at a compression ratio of 20:1 with the default encoder parameters from Section 7.3. For the decoder, we used an FED scheme with 5 nonlinear updates and 20 iterations per cycle for all tested initialisation strategies.

In Table 7.1 the deviation of the inpainting results from the reference solutions are displayed for several initialisation strategies. We measure this deviation in terms of the mean square error (MSE) between the reference reconstruction and the decoder reconstructions with different initialisation strategies.

Using a flat initialisation is completely infeasible for video playback. An average error of 461.46 suggests severe artefacts in many frames. In contrast, the reuse heuristic already decreases the average MSE to 0.20. In fact, 91.42% of the frames have a reference MSE below 0.5. In practice, they are visually indistinguishable from the reference solution. We also compare against our smart initialisation strategy, which is mainly used to avoid the creation of local artefacts in worst case scenarios as described in Section 7.2. The results in Table 7.1 also demonstrate that

Table 7.1.: Deviation of inpainting results from the corresponding reference reconstructions. Different initialisation strategies with a Fast Explicit Diffusion solver are compared in respect to the reference MSE on the movie *Nosferatu*. The default parameters from Section 7.3 were used for encoding. The maximum error is the same for reuse and smart initialisation due to the application of the t_{ref} threshold (see Section 7.3). For flat initialisation, the high maximum error is an outlier that results from ill-posed frames, where the average of the mask points is a bright grey and homogeneous black areas have to be reconstructed. Note that the main benefit of smart initialisation is the removal of local artefacts. Nevertheless, there is also an improvement in the average error.

Strategy	Flat (FED)	Reuse (FED)	Smart (FED)
Maximum Error	16788.51	4.99	4.99
Average Error	461.46	0.20	0.19
Error < 0.5	1.39%	91.42%	91.60%
Error < 1	2.51%	95.31%	95.68%
Error < 5	4.91%	100%	100%
Error < 10	6.94%	100%	100%

smart initialisation does not introduce significant negative side effects. On the contrary, the average error is decreased by 5%.

For the reuse and smart initialisation strategies, there are still some frames with an MSE close to the GOP reference threshold $t_{\text{ref}} = 5$. However, their number is relatively small and the visual difference is hard to spot at 25 frames per second.

In total, the results show that real time playback of R-EED-encoded videos is possible on consumer hardware. In particular, no compromises have to be made in regard to encoding quality in order to achieve a good approximation of the reference solution. Due to smart initialisation, no additional data has to be stored in problematic regions.

7.6 Conclusion

> *Recognizing isn't at all like seeing;*
> *the two often don't even agree.*
>
> — **Sten Nadolny**
> (Writer)

The real-time decoding capabilities of our video player framework demonstrate that even though PDE-based methods are in general more computationally intensive than transformation-based methods, a clever use of available resources allows surprising performance on consumer hardware. It does not solve the entire problem of immediacy, yet: The initialisation strategy is not directly transferable to image compression.

However, we use the parallelisation on the GPU in Chapter 8 to improve compression and decompression speed significantly. In particular, the experiences for fast inpainting that the proof of concept for video decompression provided us suffice to perform experiments on image databases for the first time. Up to this point, an evaluation was only feasible on selected test images due to runtime constraints. In this sense, the results of this chapter have not only provided a first glance at what is possible in video compression, but have also advanced image compression significantly on its way to practical viability.

Of course, from the point of view of immediacy, the future goal in image compression has to be a synchronous codec, where the difference between compression and decompression time is small. For video compression, a longer compression time is acceptable in a broadcasting context where the video is compressed once and then distributed to many recipients (for example on DVDs, Blurays or via an online video platform).

Finally, the codec that we have proposed in this chapter can act as a basis for future PDE-based methods that aim to be qualitatively competitive to transform-based coders. They can use a similar GOP structure and caching system to achieve real-time playback and, in addition, exploit temporal redundancies.

Publication Info

The video compression codec was published by Peter, Schmaltz, Mach, Mainberger, and Weickert [221] in the Journal of Visual Communication and Image Representation. In contrast to this chapter, the paper contains also a progressive mode for PDE-based compression by Schmaltz, Mach, and Mainberger and the region of interest coding from Chapter 8.1.

8 Perceptive Coding in PDE-based Image Compression

> *In this treacherous world*
> *Nothing is the truth nor a lie.*
> *Everything depends on the color*
> *Of the crystal through which one sees it.*
>
> — **Pedro Calderón de la Barca y Henao**
> (Dramatist)

In the previous chapters we have dealt with a purely quantitative view of diffusion-based compression. So far, our goal has always been to achieve reconstructions that are close to the original image w.r.t. the mean squared error (MSE). However, for many practical applications, it is more important how a human observer perceives the compressed image than how closely it approximates the ground truth.

Our Contributions. We address perceptive coding with two distinct new features for diffusion-based compression. First, we present a simple but powerful algorithm for region of interest coding, that allows users to mark important image regions that should be reconstructed more accurately than the other areas. This algorithm can be applied universally to all compression algorithms in this thesis. Our second contribution is a colour compression codec. Human perception values brightness contrast higher than colour contrast. In order to exploit this fact and simultaneously decrease the overall reconstruction error, we introduce a new diffusion-based colourisation technique. We can then store a full colour image as a monochromatic picture with a small amount of additional colour information. Colourisation recovers the full image during decompression.

Organisation of the Chapter. Since this chapter covers two different types of perceptive coding, we discuss related work in the respective subsections. Section 8.1 covers region of interest coding with a short introduction, a description of our new method in Section 8.1.1, and experiments in Section 8.1.2. For colour compression in Section 8.2, we first propose our new luma-guided colourisation method in Section 8.2.2 and evaluate it experimentally in Section 8.2.3. Section 8.2.4 is dedicated to our luma-preference mode for diffusion-based compression. Finally, we conclude the chapter in Section 8.3 with a summary and an outlook on future work.

8.1 Region of Interest Coding

> *If you just focus on the smallest details,*
> *you never get the big picture right.*
>
> — **Leroy Hood**
> (Biologist)

All the compression algorithms discussed so far have two things in common: they are optimisation-driven, and they try to minimise the overall reconstruction error. Strategies like the subdivision scheme of R-EED from Chapter 3.3 or the stochastic approaches from Chapter 5 actually try to equalise the error in different regions of the image: they store more known data where the local error is high.

However, a homogeneous distribution of the error over the whole image is not always desired. For example in medical imaging, it is important to ensure a high reconstruction quality in image regions that are relevant for diagnostic purposes, while other regions can contain more errors. In the following, we introduce a weighting to the error computation in the specific case of R-EED. However, this particular approach to region of interest (ROI) coding could also be applied to stochastic sparsification or any other optimisation-driven method.

Related Work. There is no direct support for region of interest coding in JPEG. Nevertheless, a simple standard compliant idea is to set additional DC coefficients to zero outside the region of interest (see Pennebaker and Mitchell [215]). JPEG2000 contains many methods for region of interest coding, for instance the bitplane-by-bitplane shift method (BbBShift) [288], the partial significant bitplane shift method (PSBShift) [171], or ROI coding through component priority (ROITCOP) [19]. Due to the large amount of approaches, we refer to Kaur [151] and Pawadshetty and Bakal [213] for an overview of the field. To the best of our knowledge, no PDE-based codec with ROI coding capabilities has been proposed so far.

8.1.1 Continuous Error Weighting

As discussed in Chapter 3.3, the R-EED encoder decides in which image region to add more known data based on a rectangular subdivision scheme: If the MSE of local reconstructions exceeds an error threshold, more points are added in this region. Our region of interest (ROI) extension allows to specify a continuous *a priori weighting* of different locations in the image. To this end, we introduce local weights $r_{i,j} \in [0,1]$, $i \in \{0,\dots,n_x-1\}$, $j \in \{0,\dots n_y-1\}$ for a discrete image of size $n_x \times n_y$. The ROI-weighted MSE for a reference image $\boldsymbol{f}$ and the corresponding reconstruction $\boldsymbol{u}$ is computed by

$$\mathrm{MSE}_{\mathrm{ROI}}(\boldsymbol{u},\boldsymbol{v}) := \sum_{i,j\in\Omega} \frac{r_{i,j}(u_{i,j}-f_{i,j})^2}{n_x n_y}. \tag{8.1}$$

Subdivision steps with the modified MSE lead to a higher density of known data in regions of interest with high weight $r_{i,j}$. Consequently, we can reconstruct these regions more accurately than areas with a low weight. Thereby, the ratio of weights between two regions of interest decides the turning point at which R-EED prefers a smaller quality gain in a high-priority region over a more substantial gain in quality in a low-priority region.

Note that ROI encoding only influences the choice of the inpainting mask by modifying the decisions in the subdivision tree. No additional information needs to be stored and the standard decoder is also applicable for ROI-encoded files. Furthermore, the error-weighting on a per-pixel basis allows to specify multiple regions of interest of arbitrary shape and relative importance. Since we only modify the error measurement, we can apply this ROI modification to any optimisation-driven algorithm that allows arbitrary error measures. For example, ROI coding can be used for the stochastic methods for exact masks and trees in Chapter 5.

8.1.2 Experiments

Figure 8.1 shows different applications of ROI coding. In medical imaging, ROI coding can be used to store image regions that are important for diagnostic reasons with a local error that is close to zero. The algorithm still reconstructs image regions with a lower priority, albeit with decreased accuracy. Their primary use is to coarsely represent the neighbourhood of high priority ROIs to help the observer with identifying the location of the subregions in respect to the whole imaged object.

Furthermore, ROI coding can provide semantic context to the compression algorithm. Optimisation-driven algorithms try to generate images with a consistent quality in all image regions. Especially for high compression ratios, this can lead to suboptimal perceptive image quality since important image structures like faces are treated equally with relatively unimportant background features. This behaviour is particularity detrimental in cases where unimportant features are costly to compress and thus the quality of the whole image is diminished (see Figure 8.1). Providing suitable ROI weights either manually or automatically, e.g. with a face recognition algorithm, can substantially improve the perceived quality.

Publication Info

ROI coding was published in by Peter, Schmaltz, Mach, Mainberger, and Weickert [221] in the Journal of Visual Communication and Image Representation.

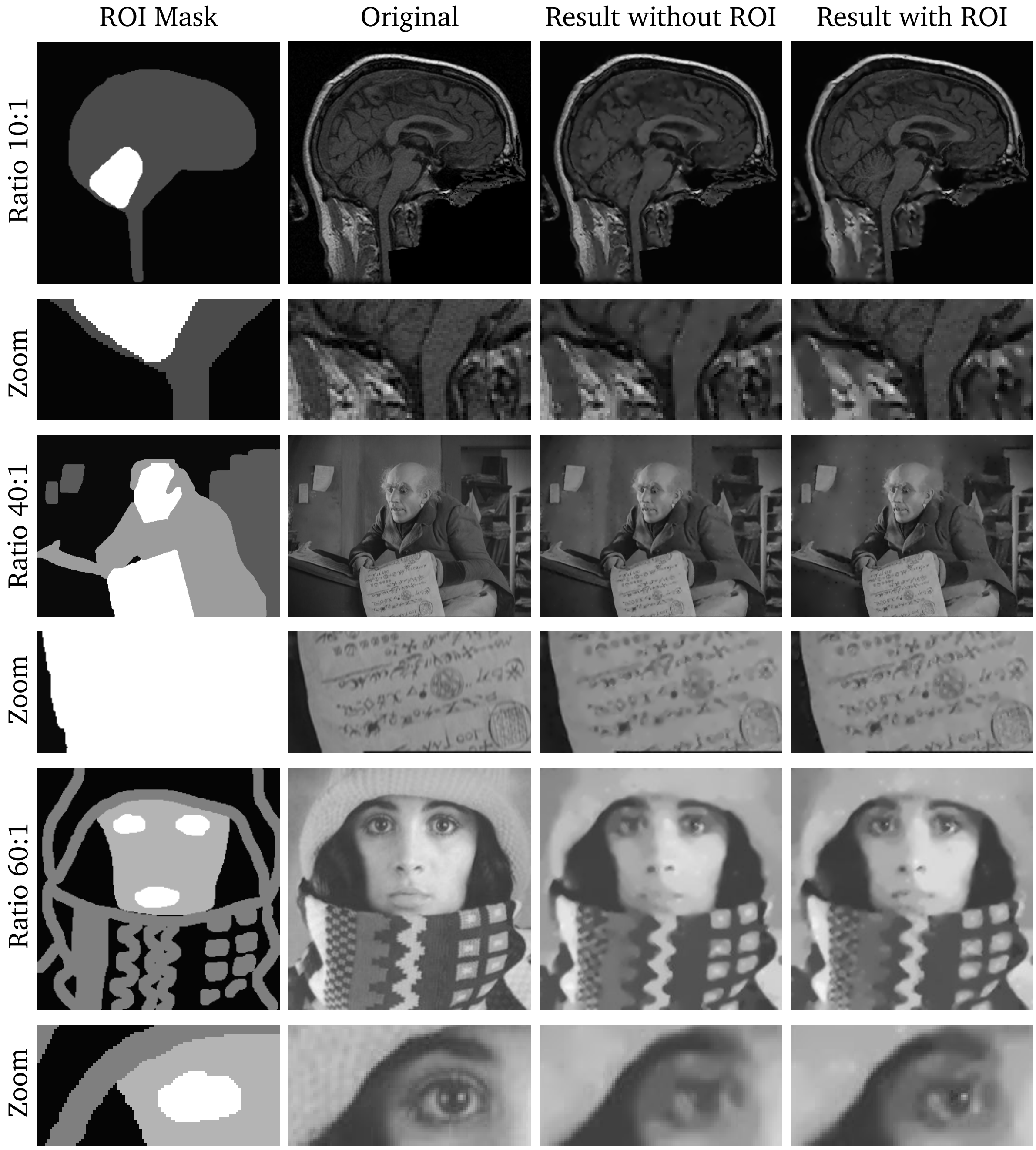


Figure 8.1.: **Reconstruction Examples with and without Region of Interest Coding for the test images *brain*, *trui*, and *nosferatu*.** The test images are ordered from top to bottom by increasing compression ratio. Bright areas in the ROI-mask have a larger weight in error computation than dark areas. For *brain* the ROI reconstruction is almost perfect in the cerebellum at the cost of reduced quality in other parts. Details such as text are reconstructed more accurately with ROI coding for *nosferatu*, while background structures are omitted. Finally, the ROI in *trui* allows a much higher perceived quality at high compression ratios due to better quality in facial features (in particular the eyes).

8.2 Compression of Colour Images with Edge-Enhancing Diffusion

> *In order to use colour effectively it is necessary to recognize that colour deceives continually.*
>
> — **Josef Albers**
> (Artist, Colour Theorist)

Colourisation is a practically relevant image processing task that has its origins in the movie industry. In the 1970's, many monochromatic films were remastered manually to include colour. Sỳkora et al. [274] provide a good overview over the history of early colourisation methods. Since this task is time-consuming and tedious, researchers aim at minimising the amount of user-interaction that is required while still allowing artists to influence the result. Levin et al. [164] have paved the way for *stroke-based* methods were the user prescribes a few colour scribbles and the colourisation algorithm fills in the missing colours.

At first glance, compression seems like an unrelated field. However, the sparse representations of images created by our diffusion-based codecs from the previous chapters are comparable to the colour scribbles in stroke-based colourisation. In Chapter 1, we have already illustrated in Figure 1.3, that R-EED [250], the current state-of-the art in PDE-based compression, is not able to beat JPEG2000 [277] on colour images. Both JPEG [215] and JPEG2000 exploit the fact that the human visual system values structure higher than colour. The different perception of monochromatic and colour contrast is well documented from a biological point of view (see e.g. Mullen [198]). Consequently, transform-based coders use colour spaces that decompose the image into brightness and colour information to compress colour channels in a coarser manner. In comparison, R-EED only allows compression in an RGB colour space [252] and does not provide the option to take into account human perception.

In regards to compression, colourisation has so far only been integrated into JPEG and JPEG2000. However, diffusion-based codecs are related much more closely to the concept of Levin et al. and are lacking a dedicated colour mode. In this chapter, we close this gap with a new continuous diffusion-framework for colourisation and a colour codec based on R-EED.

Our Contributions. Since the successful method of Levin et al. has been formulated purely in a discrete setting, we present a new continuous framework for colourisation. We consider the problem in YCbCr space where the luma (brightness) channel Y is given by the greyscale input image. Our novel *luma-guided* diffusion adapts to the structure of the brightness channel and propagates the user-specified colour scribbles in the chroma (colour) channels to the missing areas. We evaluate four different diffusion models from our general framework, namely linear isotropic and edge-enhancing diffusion, as well as two newly proposed higher-order counterparts to the aforementioned models. We show that the method of Levin et al. is closely related to higher-order isotropic diffusion. Moreover, we demonstrate with experiments on well-known test images that our anisotropic models outperform isotropic methods significantly.

For compression, we combine our luma-guided diffusion with the R-EED codec to a *luma preference (LP) mode*. It relies on the core idea that the structural information in the luma channel is more important for the human visual system than the colour components. Therefore, this channel should be stored with higher accuracy than the chroma channels Cb and Cr. To this end, we dedicate a large part of our file size budget to the luma channel. The luma-guided diffusion profits from the accurate structure in the brightness channel and allows to reconstruct the chroma channels from a smaller amount of known data.

Moreover, we use a more efficient implementation than the original R-EED codec. Building on our results for real-time diffusion in Chapter 7, it uses different solvers and parallelisation on the GPU. This allows us to conduct experiments on a real-world image database [83] for the first time in diffusion-based compression.

8.2.1 Related Work

Colourisation. Following the taxonomy of Luan et al. [176], we classify colourisation methods as *stroke*- or *example*-based. In stroke-based methods [135, 147, 154, 164, 176, 231, 313], the user manually specifies a small amount of colour scribbles on a single image or a sequence of images. In contrast, *example*-based algorithms [142, 274, 276, 306] require one or multiple fully coloured images that are similar to the greyscale image to be colourised. This approach is particularly useful for video sequences and can be combined with stroke-based approaches: A single frame can be coloured with scribbles and serve as an example for the remaining frames. Since we focus on single images, we only consider the stroke-based approach. For a review for colourisation by example, we refer to Luan et al. [176].

In this chapter, we establish a close relationship of our diffusion framework to the work of Levin et al. [164]. They minimise a discrete energy that relies on a simple assumption: Neighbouring pixels with similar intensities also have similar colours. To this end, they first transform the image to YUV space to obtain a structure-colour-decomposition similar to YCbCr. Then, they minimise the difference of each unknown colour pixel in YUV space to a weighted average of its 3×3 neighbourhood. The weights are determined by the Y channel.

Several other contributions directly extend the concepts of Levin et al. [164]. Huang et al. [135] modify the weighting function and, in order to stop colour propagation across brightness edges, they compute thin, closed edges with a Sobel operator and multiple postprocessing steps. With an approach inspired by total variation models, Kang and March [147] deviate further from the ideas of Levin et al. However, they still rely on energy minimisation. Lischinski et al. [168] go beyond the original task of colourisation and extend the method of Levin et al. to a wide variety of other tasks such as tone mapping or hight dynamic range imaging. In a similar direction, Hua et al. [132] interpret the approach of Levin et al. from a gradient domain perspective. They propose a framework with edge-preserving constraints that includes colourisation among other tasks.

Konushin and Vezhnevets [154] employ a biologically motivated algorithm that models the averaging of colours by the spread and mutation of bacteria over cells that correspond to the image pixels. There are also various non-local approaches for colourisation. Yatziv and Sapiro [316] work in YCbCr space. They obtain the missing colour in each pixel as a weighted average of known colours according to their respective geodesic distance in the luma channel. Luan et al. [176] group pixels locally by intensity and globally by texture similarity before they propagate colour. A mapping between brightness and colour values is obtained by Quang et al. [231] by means of a machine learning approach. Yang [313] propagates colour information with a recursive bilateral filter. For the specialised class of monochromatic manga comics, Qu et al. [230] use clustering according to texture similarity and evolving level sets.

The higher-order approaches that we consider in Section 8.2.2 are very rare in the literature. To our best knowledge, only Greer et al. [112] have proposed a similar, but nonlinear model in the context of heat flow on surfaces.

Compression. In the second part of this chapter, we propose a new diffusion-based compression approach. We rely on a sophisticated extension of the R-EED codec by Schmaltz et al. [250]. We have discussed this method in more detail in Chapter 3.3. So far, there are no dedicated colour codecs with diffusion, but there is a variety of other specialised compression methods. They focus for example on cartoons [183], depth maps [103, 127, 166], 3-D data [218], or texture [220]. Chapter 3.2 provides a full review of diffusion-based compression.

Our colour compression codec is inspired by the treatment of colour in the transform-based coders JPEG [215] and JPEG2000 [277]. Both of them exploit the properties of the human visual system in order to improve the perceived fidelity of compressed colour images. JPEG performs a subsampling of the chroma channels in YCbCr space, while JPEG2000 omits fine-scale wavelet coefficients for the colour components of YUV space.

With our method, we go one step further since we compensate the reduced amount of known data for the colour channels by reusing structural information from the brightness component. Similarly, some transformation-based coders exploit colourisation ideas for compression. Most of them rely on the method of Levin et al. [164]. In [129], Horiuchi and Tominagaan use an extended version of the method of Levin et al. in combination with JPEG to restore colour information from samples in CIELAB space. Cheng and Vishwanathan [60] investigate a machine learning approach to the problem. They interpret the approach of Levin et al. as a learning algorithm and incorporate a modified version into JPEG. Furthermore, variations of a colourisation method based on Markov random fields [207] have been applied in a postprocessing step to JPEG [208] and JPEG2000 [206]. In a more broadly related approach that goes beyond colour image compression, Förster et al. [95] also employ the concept of reusing structural information of a guidance image: They compress depth maps with compressed sensing and exploit information of the corresponding RGB-image.

8.2.2 From Inpainting to Colourisation

Recap: Inpainting RGB-Colour Images.

Diffusion inpainting on colour images, as it is used in R-EED, considers a coupled model. For an image $\boldsymbol{f} = (f_R, f_G, f_B)^\top$ with the RGB channels $f_c : \Omega \to \mathbb{R}$, $c \in \{R, G, B\}$, we assume the full colour data is given at each pixel of the inpainting mask $K \subset \Omega$. As for denoising of colour images in Chapter 4, we consider the PDE

$$\partial_t u_c = \operatorname{div}(\boldsymbol{D}(\boldsymbol{J}_{RGB})\boldsymbol{\nabla} u_c), \quad c \in \{R, G, B\} \tag{8.2}$$

with reflecting boundary conditions at the image boundaries $\partial\Omega$ and fixed known data on K. The diffusion tensor adapts to the local image geometry according to Di Zenzo's structure tensor for colour images [75]:

$$\boldsymbol{J}_{RGB} := \sum_{c \in \{R,G,B\}}^{3} \boldsymbol{\nabla} u_{c,\sigma} \boldsymbol{\nabla} u_{c,\sigma}^\top. \tag{8.3}$$

However, this time, the known data is fixed on K and we consider the steady state $t \to \infty$ of Eq. 8.2 to obtain the reconstruction on $\Omega \setminus K$. As in Chapter 4, we can penalise the eigenvalues of the joint structure tensor $\boldsymbol{J}_{RGB}$ in different ways to obtain isotropic or anisotropic diffusion models.

Colourisation Models

In contrast to inpainting, a full original grey value image is known for colourisation. In addition, a user can specify some colour information at arbitrary positions. In practice, users often draw some freehand colour "scribbles" into the image (see Fig. 8.3 (a)).

The formulation of inpainting in RGB space from Eq. 8.2 is not adequate for an elegant mathematical description of this situation. For colourisation, we need a clear separation of the brightness information from the original image and the newly added colour data. Therefore, we use the YCbCr colour space instead. It separates the image data into intensity information in the *luma channel* Y and colour information in the *chroma channels* Cb and Cr; see e.g. Pratt [228]. We use the following transform of an RGB image $\boldsymbol{f} = (f_R, f_G, f_B)$ to YCbCr space

$$\begin{pmatrix} f_Y \\ f_{Cb} \\ f_{Cr} \end{pmatrix} = \begin{pmatrix} 0 \\ 127.5 \\ 127.5 \end{pmatrix} + \boldsymbol{T} \begin{pmatrix} f_R \\ f_G \\ f_B \end{pmatrix}, \tag{8.4}$$

$$\boldsymbol{T} := \begin{pmatrix} 0.2990 & 0.5870 & 0.1140 \\ -0.1687 & -0.3313 & 0.5000 \\ 0.5000 & -0.4187 & -0.0813 \end{pmatrix}. \tag{8.5}$$

Please note that "luma" is an engeneering term from the original YCbCr definition for TV formats. For the sake of consistency with the YCbCr notation we use this term to refer to the channel that contains our abstract, device-agnostic brightness information. We do not consider photometric concepts in this context.

With this transformation, we can now adequately describe the colourisation task: We want to obtain a colourised version $\boldsymbol{u} : \Omega \to \mathbb{R}^3$ of a greyscale image $f : \Omega \to \mathbb{R}$. In YCbCr space, f corresponds to the Y channel of u, which yields $u_Y = f$ on the entire domain Ω. On a subset $K \subset \Omega$, colour scribbles $u_{Cb} = f_{Cb}$ and $u_{Cr} = f_{Cr}$ are given in addition. Our task is now to inpaint the missing colours in $\Omega \setminus K$ in the channels u_{Cb} and u_{Cr}.

Again, our processes should be able to reconstruct sharp edges accurately from small amounts of data. Fortunately, chroma edges mostly coincide with luma edges. Therefore, we do not use a joint structure tensor (see Eq. (8.3)) to adapt our colourisation locally to the image structure. Instead, we consider a *luma tensor*

$$\boldsymbol{J}_Y := \boldsymbol{\nabla} u_{Y,\sigma} \boldsymbol{\nabla} u_{Y,\sigma}^\top \tag{8.6}$$

which describes only the known image structure in the brightness channel. Now we can formulate the colourisation in YCbCr space as a variational problem. We impose a smoothness constraint to the missing parts of the chroma channels. For a given channel $c \in \{Cb, Cr\}$, we minimise the energy functional

$$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} \left(\boldsymbol{\nabla}^\top u_c \boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c \right) d\boldsymbol{x}. \tag{8.7}$$

Variational calculus tells us that a channel u_c that minimises the energy from Eq. 8.7 must fulfil the Euler-Lagrange Equation

$$0 = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c), \quad c \in \{Cb, Cr\} \tag{8.8}$$

Table 8.1.: Overview of our Diffusion-based Colourisation Methods.

model	energy	diffusion PDE
isotropic	$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} g(\lvert \boldsymbol{\nabla} u_Y \rvert^2) \, \lvert \boldsymbol{\nabla} u_c \rvert^2 \, d\boldsymbol{x}$	$\partial_t u_c = \text{div}(g(\lvert \boldsymbol{\nabla} u_Y \rvert^2) \boldsymbol{\nabla} u_c)$
bi-isotropic	$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} \left(\text{div}(g(\lvert \boldsymbol{\nabla} u_Y \rvert^2) \boldsymbol{\nabla} u_c)\right)^2 d\boldsymbol{x}$	$\partial_t u_c = \text{div}(g(\lvert \boldsymbol{\nabla} u_Y \rvert^2) \boldsymbol{\nabla}(\text{div}(g(\lvert \boldsymbol{\nabla} u_Y \rvert^2) \boldsymbol{\nabla} u_c)))$
EED	$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} \boldsymbol{\nabla}^\top u_c \boldsymbol{D}(\boldsymbol{J}_Y) \, \boldsymbol{\nabla} u_c \, d\boldsymbol{x}$	$\partial_t u_c = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c)$
Bi-EED	$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} \left(\text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c)\right)^2 d\boldsymbol{x}$	$\partial_t u_c = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla}(\text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c)))$

on $\Omega \setminus K$. This corresponds to the steady state of the diffusion process

$$\partial_t u_c = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c), \quad c \in \{Cb, Cr\} \tag{8.9}$$

with Dirichlet boundary conditions $u_c = f_c$ on K.

Since the propagation is steered by the brightness information in the luma tensor $\boldsymbol{J}_Y$ this *luma-guided* diffusion can also reconstruct edges in the chroma channels. Note that the luma tensor has the eigenvectors $\boldsymbol{v}_1 = \boldsymbol{\nabla} u_{Y,\sigma} / \lvert \boldsymbol{\nabla} u_{Y,\sigma} \rvert$ and $\boldsymbol{v}_2 = \boldsymbol{\nabla} u_{Y,\sigma}^{\perp} / \lvert \boldsymbol{\nabla} u_{Y,\sigma} \rvert$ with the corresponding eigenvalues $\mu_1 = \lvert \boldsymbol{\nabla} u_{Y,\sigma} \rvert^2$ and $\mu_2 = 0$. As in the previous sections, we choose the eigenvectors of $\boldsymbol{D}(\boldsymbol{J}_Y)$ to be $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$.

If we just want to inhibit diffusion at luma edges, we can set the eigenvalues of $\boldsymbol{D}$ to $\lambda_1 = \lambda_2 = g(\lvert \boldsymbol{\nabla} u_{Y,\sigma} \rvert^2)$ with the Charbonnier diffusivity g from Eq. 3.20 in Chapter 3.1. This leads to a *linear* isotropic inpainting model with scalar diffusivity instead of a diffusion tensor. Setting $\lambda_2 = 1$ allows full diffusion along luma edges, while the diffusion across them is still restricted. Thus, our model also allows luma-guided edge-enhancing anisotropic diffusion.

Finally, we can also derive higher-order colourisation models from a functional of the form

$$E(u_c) = \frac{1}{2} \int_{\Omega \setminus K} (\text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c))^2 d\boldsymbol{x}. \tag{8.10}$$

It leads us to a novel higher-order diffusion process of the form

$$\partial_t u = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla}(\text{div}(\boldsymbol{D}(\boldsymbol{J}_Y) \boldsymbol{\nabla} u_c))). \tag{8.11}$$

Again, by choosing the eigenvalues of the diffusion tensor $\boldsymbol{D}(\boldsymbol{J}_Y)$ we can obtain both isotropic and anisotropic processes. In accordance to the biharmonic operator ($\boldsymbol{D}(\boldsymbol{J}_Y) = \boldsymbol{I}$) we denote these two diffusion models as *bi-isotropic* and *Bi-EED*. Table 8.1 provides an overview over the four energies and their corresponding diffusion PDEs.

Relation to the Method of Levin et al. [164]

In the following, we will show that the method of Levin et al. is closely related to the variational formulation of our higher-order isotropic model from the previous section. In order to obtain a discrete energy from the continuous bi-isotropic functional in Table 8.1, we consider a standard finite-difference approximation [294] of the divergence term with equal grid size h in each direction:

$$\begin{aligned} \operatorname{div}(g(|\boldsymbol{\nabla} u_Y|^2) \approx \frac{1}{h}\Big(& \frac{g_{i+1,j}+g_{i,j}}{2}\,\frac{u_{c,i+1,j}-u_{c,i,j}}{h} && (8.12)\\ -& \frac{g_{i,j}+g_{i-1,j}}{2}\,\frac{u_{c,i,j}-u_{c,i-1,j}}{h} && (8.13)\\ +& \frac{g_{i,j+1}+g_{i,j}}{2}\,\frac{u_{c,i,j+1}-u_{c,i,j}}{h} && (8.14)\\ -& \frac{g_{i,j}+g_{i,j-1}}{2}\,\frac{u_{c,i,j}-u_{c,i,j-1}}{h}\Big). && (8.15) \end{aligned}$$

This detailed notation of the discretisation emphasizes the averaging of diffusivities and the derivative approximations, but it can also be expressed in a more compact way. Let $\mathcal{N}(\boldsymbol{r})$ denote the set of indices belonging to the 3×3 neighbourhood of the image point $\boldsymbol{r} \in \Gamma = \{0, ..., n_x-1\} \times \{0, ..., n_y-1\}$ in the discrete image domain of size $n_x \times n_y$. Furthermore, we define the neighbourhood weighting function

$$w_{\boldsymbol{r},\boldsymbol{s}} = \begin{cases} \frac{1}{2}\frac{g_{\boldsymbol{s}}}{h^2} & (\boldsymbol{r} \neq \boldsymbol{s}), \\ 2\frac{g_{\boldsymbol{s}}}{h^2} & (\boldsymbol{r} = \boldsymbol{s}). \end{cases} \tag{8.16}$$

With our new notation we can write the discrete energy that corresponds to Eq. (8.15) as

$$E(u_c) = \sum_{r \in \Gamma \setminus K} \left(\left(\sum_{s \in \mathcal{N}(\boldsymbol{r})} w_{\boldsymbol{r},\boldsymbol{s}} \right) u_{c,\boldsymbol{r}} - \sum_{s \in \mathcal{N}(\boldsymbol{r})} w_{\boldsymbol{r},\boldsymbol{s}} u_{c,\boldsymbol{s}} \right)^2 . \tag{8.17}$$

The discrete energy from Eq. 8.17 has exactly the same form as the one of Levin et al. Only the weights $w_{\boldsymbol{r},\boldsymbol{s}}$ are not necessarily the same. Levin et al. choose them according to a normalised correlation between the intensity values $u_Y(\boldsymbol{r})$ and $u_Y(\boldsymbol{s})$, while we employ a diffusivity.

8.2.3 Colourisation Experiments

In the following, we want to evaluate the performance of our four methods luma-guided isotropic and bi-isotropic diffusion, EED, and Bi-EED against the reference implementation of the method of Levin et al. In order to achieve a quantitative evaluation, we colourise the brightness channel of the well known test images *house*, *landscape*, and *mandrill*, as well as *fish*, an image from the Berkeley database [188]. We only select a very small amount of colour scribbles or regular samples in Figure 8.3 and Figure 8.4. After colourisation, we compute the mean-squared error (MSE) in respect to the original colour images.

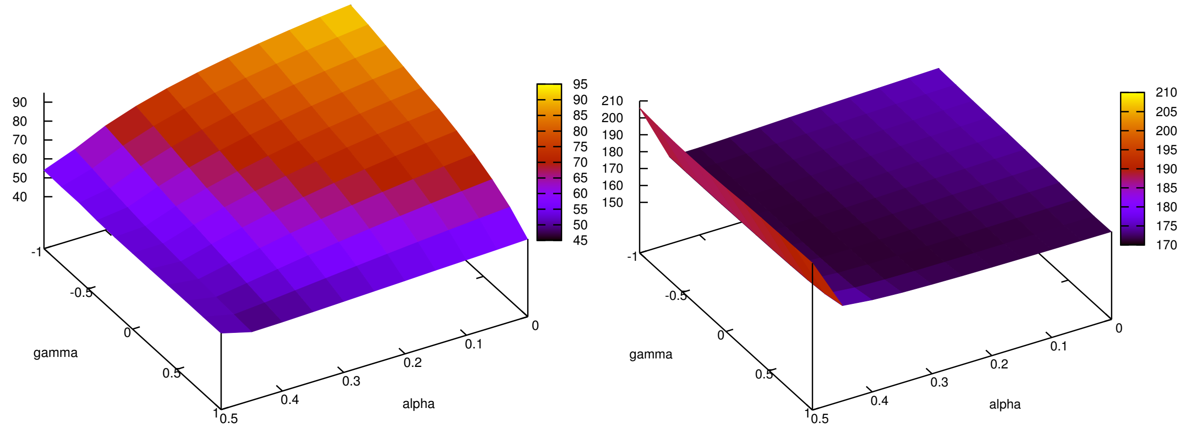


Figure 8.2.: **Error Dependencies on the Discretisation Parameters α and γ on the Test Images *fish* (left) and *mandrill* (right).** Depending on the image content and known data, the discretisation can have a large impact or only affect the reconstruction if border cases are used. The contrast parameter $\lambda = 0.01$ was also optimised for the overall smallest error.

For our implementation, we use the finite difference framework of Weickert et al. [299]. We consider the evolution of the parabolic PDEs from the previous section and solve the inpainting problem iteratively with fast explicit diffusion (FED) [114] combined with a coarse-to-fine initialisation. We stop the iterative scheme as soon as the norm of the residual has decreased by a factor 10^{-5}.

Experimentally, we have determined that a small contrast parameter (λ in Eq. 3.20 from Chapter 3.1) seems to consistently yield the best results for all four of our diffusion models. Thereby, the contrast parameter is fixed to $\lambda = 0.01$ for all results shown in this section. Note that for our later experiments with compression, where the original luma component is only approximated, this does not always hold.

hFurthermore, we have also investigated the influence of the discretisation parameters α and γ on the colourisation results. More information on these parameters can be found in Eqs. 3.44–3.47 and the accompanying explanations in Chapter 3.1. Optimal parameters and the gain from optimisation of the discretisation vary from image to image. In the case of the image *fish*, the error varies from 92.38 ($\alpha = 0$, $\gamma = -1$) to 48.61 ($\alpha = 0.4$, $\gamma = 1$) (see Fig. 8.2). However, this influence depends a lot on the image content. For example, in the case of the image *mandrill*, most parameter choices yielded almost the same reconstruction error. For the following experiments, we choose the optimal α and γ for the respective images.

As Tab. 8.2 reveals, the isotropic PDEs perform slightly worse than the method of Levin et al., but the anisotropic models yield by far the best results. As the zoom-in into the test image *fish* in Figure 8.5 reveals, the edge-enhancing properties of anisotropic diffusion allow to reconstruct areas with sharp colour contrast accurately: The other models wrongly propagate red colour into the white stripes of the fish, an effect that is often called *colour bleeding*. Similarly, they

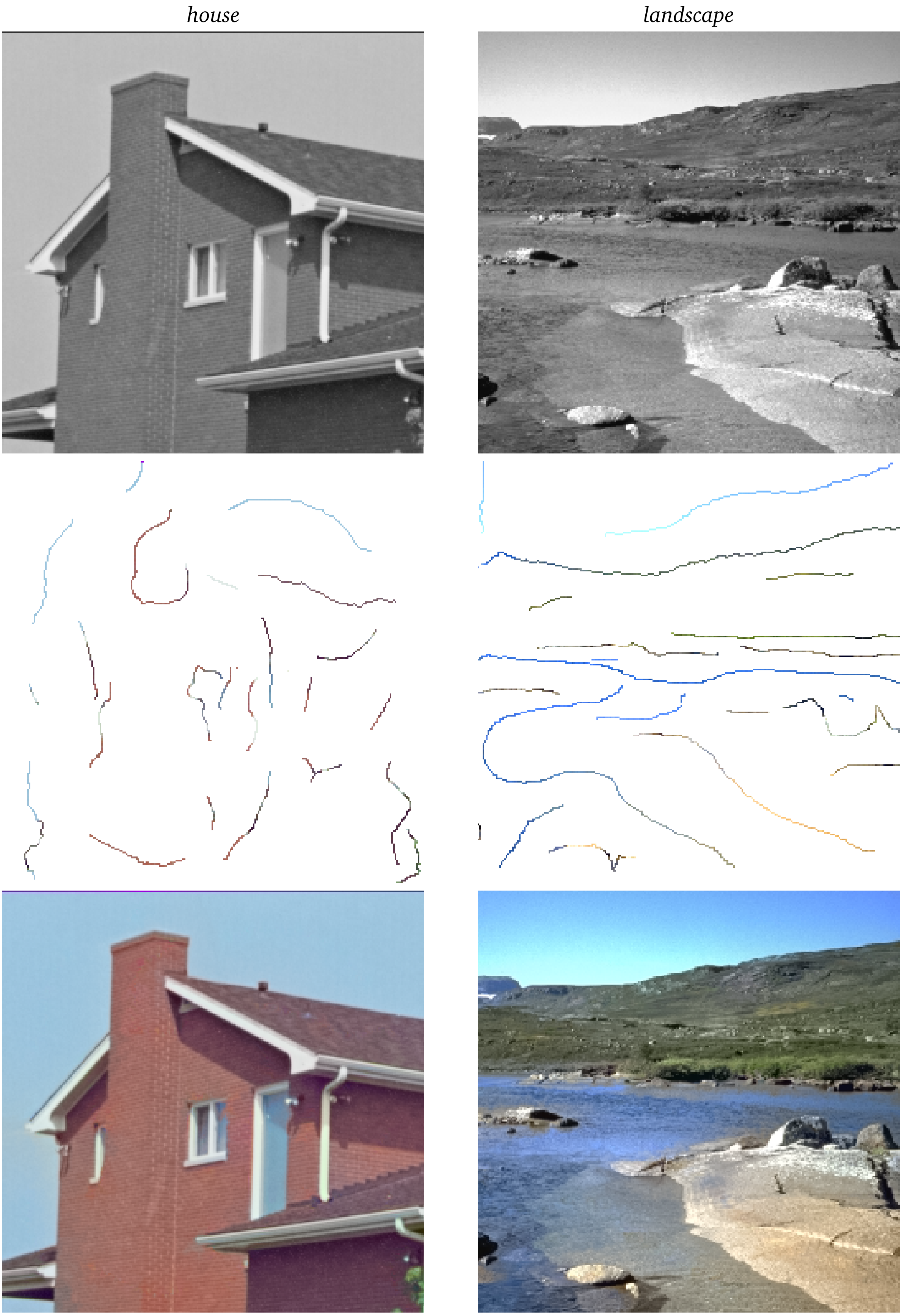


Figure 8.3.: **Colourisation with Bi-EED** for *house* and *landscape*. For each image, we show the greyscale version (top), the colour scribbles (middle) and the colourised images (bottom).

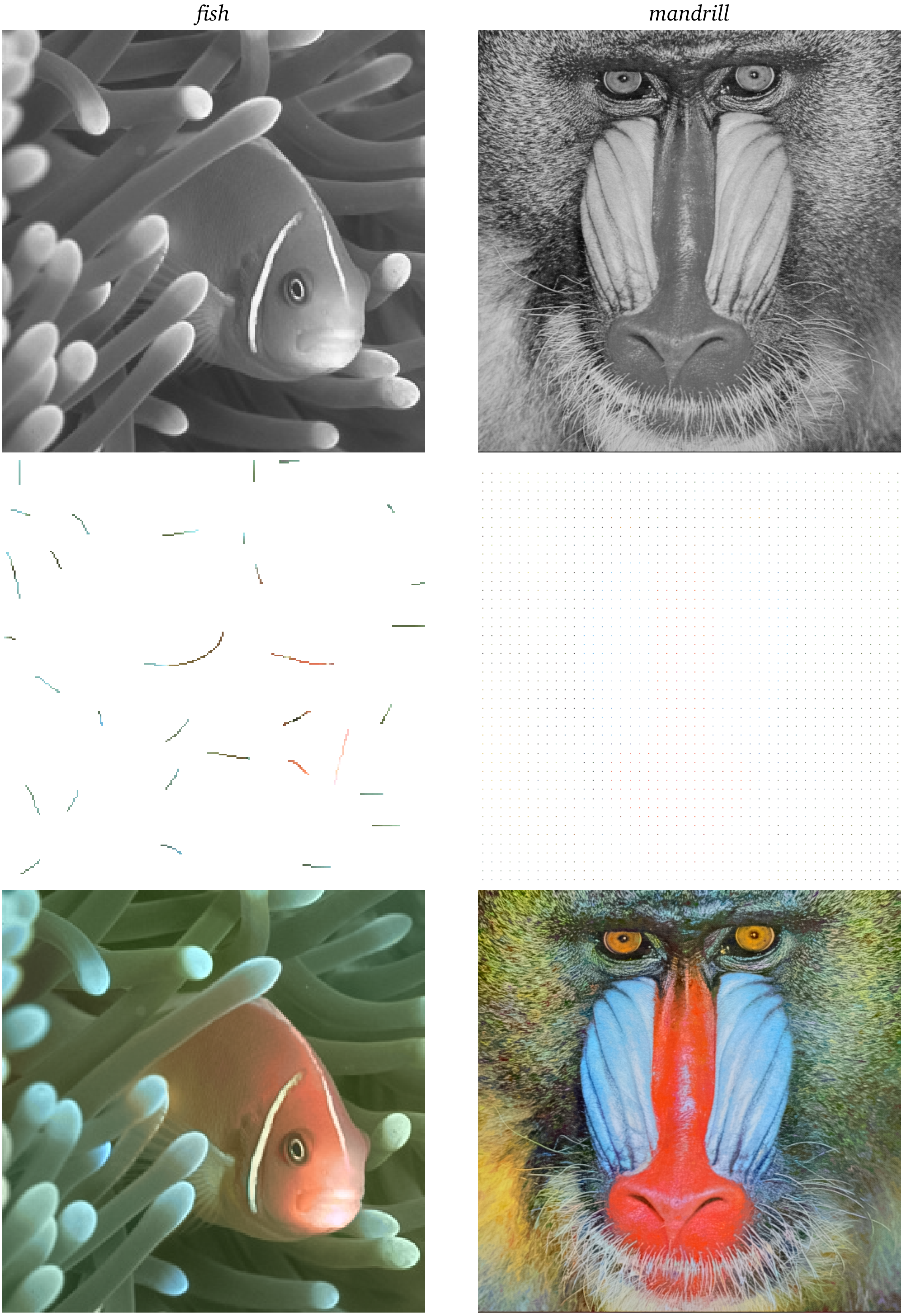


Figure 8.4.: **Colourisation with Bi-EED** for *fish* and *mandrill*. For each image, we show the greyscale version (top), the colour scribbles (middle) and the colourised images (bottom).

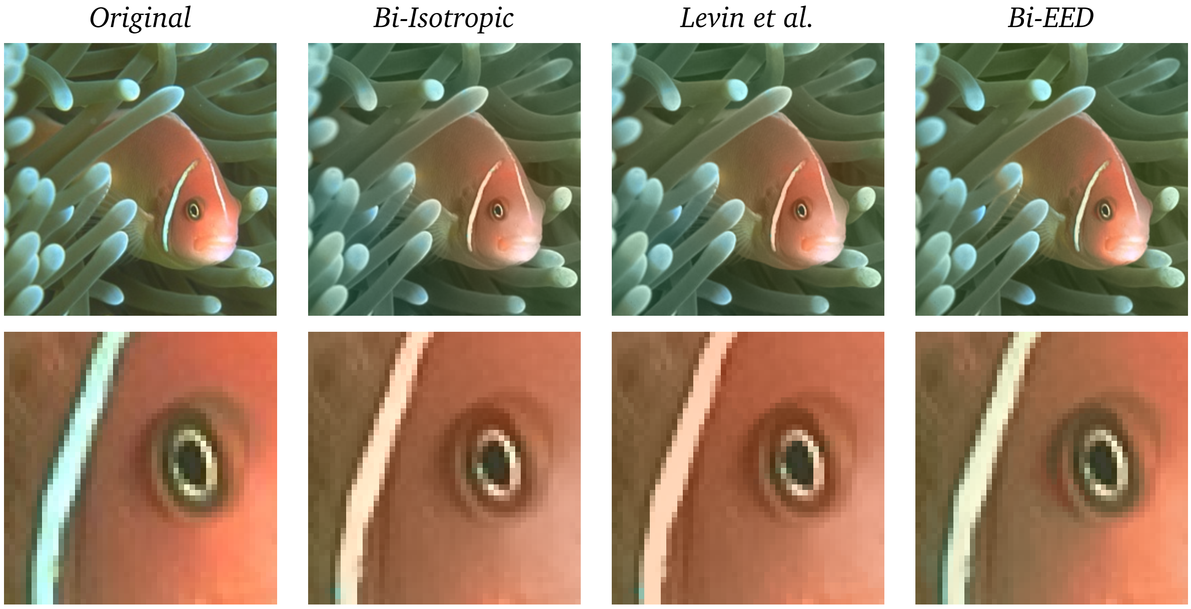


Figure 8.5.: **Comparison of Different Colourisation Methods for the Test Image *fish*.** Since the structure in the luma channel is identical in all images, the differences are hard to spot for the human visual system. However, zooming into the images reveals significant differences. Perceptually, the bi-isotropic model and the model of Levin et al. yield similar results. Bi-EED reproduces the high contrast differences of the eye and the white stripe significantly better than both competitors. The isotropic and EED results are omitted here, since they cannot be visually distinguished from the other methods.

Table 8.2.: Comparison of colourisation techniques by MSE.

Algorithm	*house*	*landscape*	*fish*	*mandrill*
isotropic	101.59	86.64	105.88	205.33
bi-isotropic	101.59	86.70	105.82	205.34
Levin et al.	75.46	82.57	88.61	194.25
EED	**57.70**	65.88	48.98	171.22
Bi-EED	58.48	**65.86**	**48.61**	**170.94**

fail to reconstruct the eye as sharply as EED or Bi-EED. The colourisation results of Bi-EED for all images can be found in Figure 8.3 and Figure 8.4.

Tab. 8.2 also demonstrates that, in general, higher-order models do not yield a significant advantage. Empirically, we found that they have an evolution that differs from their lower-order counterpart, but yield almost the same steady state. Therefore, isotropic diffusion and EED should be preferred for this specific task since they converge significantly faster.

All of the aforementioned experiments are artificial in the sense that a colour ground truth is available. In real-life applications of colourisation, only the greyscale image is known and colour must be specified manually. In Figure 8.6 we present a colourisation of an authentic

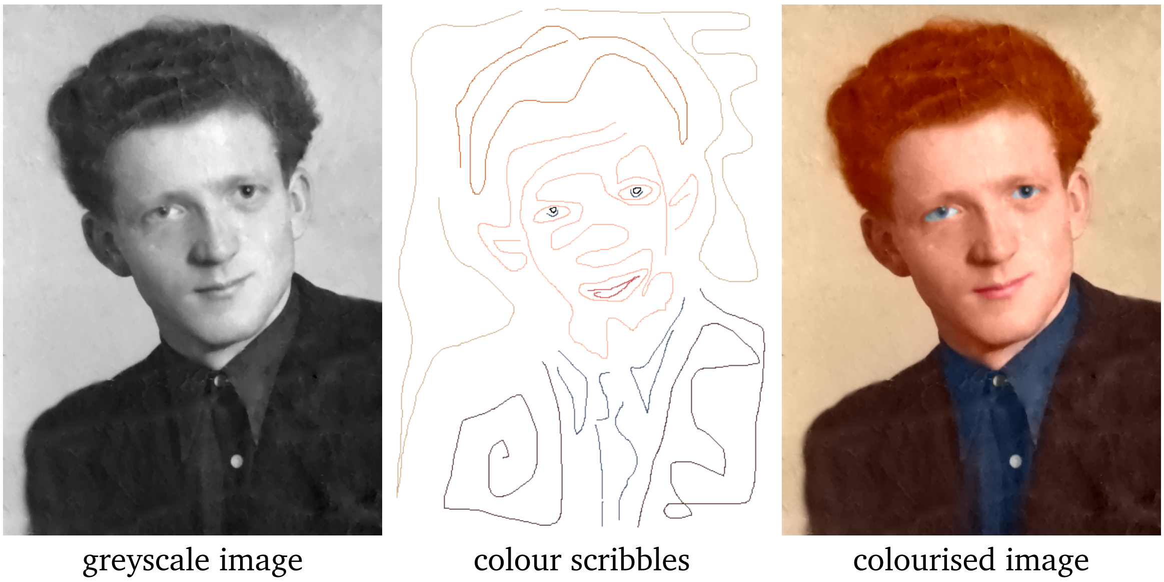


Figure 8.6.: **Authentic Image Colourisation. Left:** Authentic photograph of the author's grandfather. Restored with EED-based inpainting in Chapter 3.1, Figure 3.3. **Middle:** Manually drawn colour scribbles. **Right:** Colourisation with EED.

greyscale photograph. Only a few, coarsely drawn colour scribbles suffice to achieve visually pleasing results.

8.2.4 Compression with Luma Preference Mode

Currently, R-EED by Schmaltz et al. [250] only supports compression of colour images in RGB space. It treats all channels equally by inpainting with the coupled multi-channel EED from Section 3.3. The locations of known data are shared between these channels. This reduces the overhead since only one inpainting mask needs to be stored, but it also implicitly encodes the image structure (such as edges) in all three channels. In the following, we use our luma-guided inpainting in YCbCr space to eliminate this redundancy.

Since we use YCbCr space, we can also exploit the fact that the human visual system is much more sensitive to errors in the luma channel than to deviations in the chroma components (see e.g. [36]). Therefore, we can increase the perceived quality by storing the luma channel with higher accuracy compared to the chroma information.

Motivated by the two observations above, we propose a *luma preference (LP) mode* for colour compression with R-EED. In order to reach our goal of accurate luma compression, we prescribe a target compression ratio $R:1$ and dedicate a larger amount of the resulting bit budget to the brightness channel. We describe this weighting of the luma and chroma components by the *LP ratio* $r \geq 1$. This free parameter expresses the file sizes s_{Cb} and s_{Cr} for colour data as multiples of the luma size s_Y:

$$s_Y = r \cdot s_{Cb} = r \cdot s_{Cr}. \tag{8.18}$$

Therefore, the desired compression ratio $R : 1$ also determines s_Y. Due to Eq. (8.18), the total size of the compressed file is $(1 + 2r)s_Y$. Consequently, for an original file size s_O, LP mode reaches the target ratio if s_Y satisfies

$$s_Y := \frac{s_O}{(1 + 2r)R}. \tag{8.19}$$

Our *channel weighting* with the LP ratio introduces perceptive coding to R-EED, acting as a counterpart to chroma subsampling in JPEG. However, this by itself only addresses one of our two goals. Instead of just improving the quality of structural information at the cost of colour accuracy, we want to exploit correlations between the channels. Even though the edges are not identical in the Y, Cb, and Cr components, a luma edge is still a good indicator for a chroma edge, as the high-fidelity colourisations in Section 8.2.3 demonstrate. To this end, we use the compressed luma channel as the grey-valued guidance image for our colourisation method from Section 8.2.2. In contrast to Eq. (8.9), we inpaint all three channels with the luma diffusion tensor. Note that we have individual known data K_c now for every channel. On $\Omega \setminus K_c$, we compute the steady state of the evolution

$$\partial_t u_c = \text{div}(\boldsymbol{D}(\boldsymbol{J}_Y)\boldsymbol{\nabla} u_c), \quad c \in \{Y, Cb, Cr\}. \tag{8.20}$$

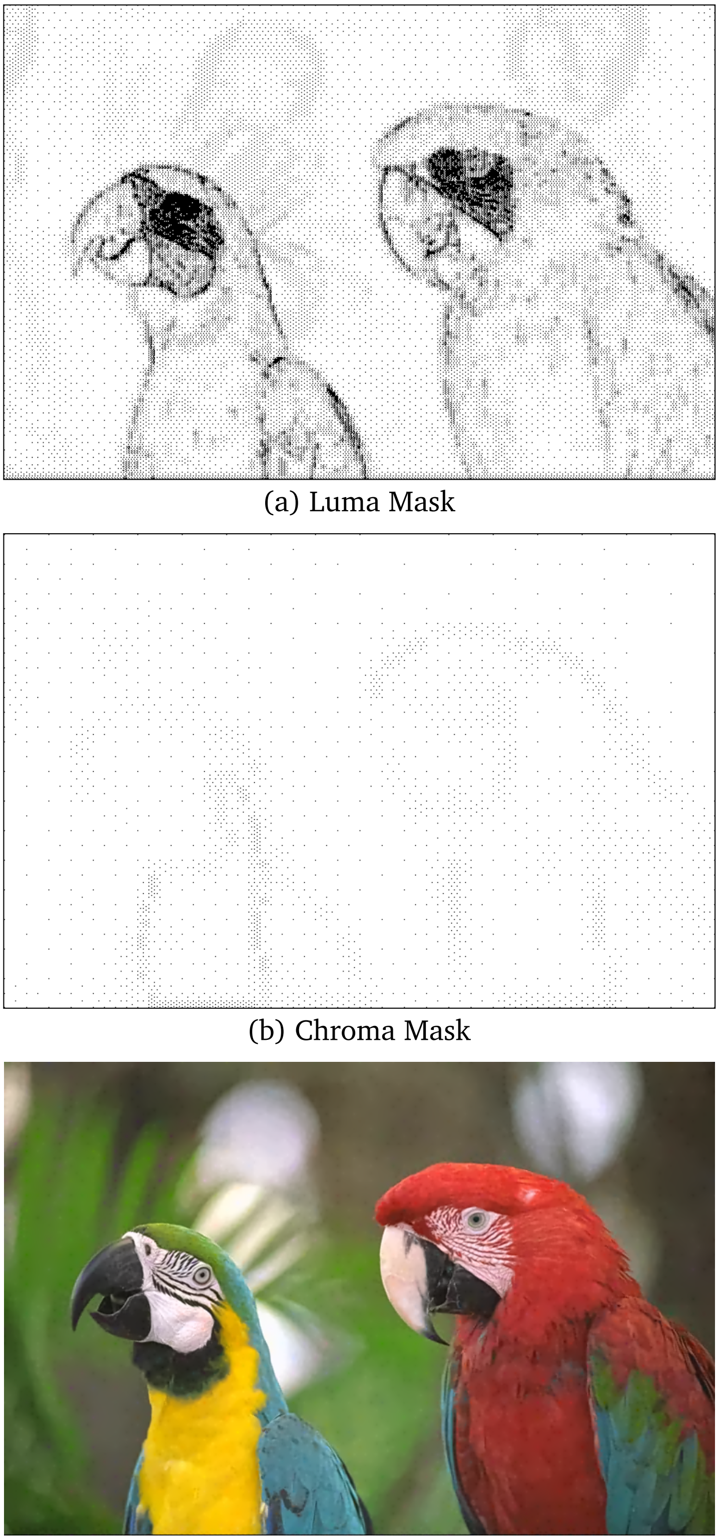
Since it offers the best combination of reconstruction quality and performance in the experiments from Section 8.2.3, we use the anisotropic diffusion tensor. Thus, we perform regular non-linear edge-enhancing diffusion in the luma channel and linear luma-guided EED in the chroma channels.

Finally, we combine the channel weighting and luma-guided diffusion to our luma preference codec. In the following, we describe its compression and decompression pipelines as well as its file structure (see Fig. 8.8 and Fig. 8.9).

Compression in LP mode consists of two sequential steps.

Step 1: Luma Compression. For the luma channel, we obtain the mask with the R-EED subdivision scheme from Section 3.3. We optimise the contrast parameter λ_Y, and the number of quantised grey-values q_Y in the luma channel for the best brightness MSE at the target file size s_Y. We approximate s_Y by compressing the positions of known data represented by a binary *luma tree* and the quantised grey values with PAQ [178]. In addition, we optimise the intensity values at the locations of the final luma inpainting mask.

Step 2: Chroma Compression. In this compression step, we also perform subdivision and optimise a contrast parameter λ_{CbCr} and a quantisation parameter q_{CbCr}. However, there are



(a) Luma Mask

(b) Chroma Mask

(c) Reconstruction

Figure 8.7.: **Masks and Reconstruction in LP Mode.** Image 23 of the Kodak database with compression ratio 60:1, LP factor $r = 0.625$. Known data clusters near edges and textured areas in the luma mask. This encodes structure with very high accuracy. For the chroma mask, we only require a few scattered points that indicate colour.

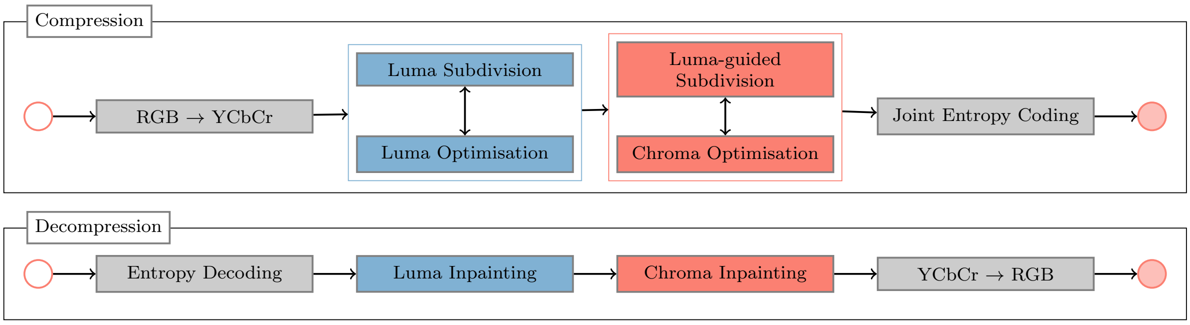


Figure 8.8.: **Compression and Decompression in LP-mode.** In both steps, luma inpainting has to be performed first such that the luma tensor is available for luma-guided inpainting.

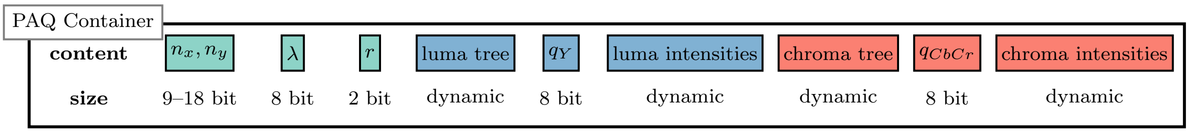


Figure 8.9.: **File Structure in LP Mode** for a $n_x \times n_y$ image with contrast parameters $\lambda_Y, \lambda_{CbCr}$, and quantised grey values q_Y, q_{CbCr}.

some important differences. We use luma-guided EED in this step, which also implies that accurate structural information is already encoded in the luma tensor. Thus, the exact position of the known data is not as important as in the first step. To reduce overhead, we therefore use a joint *chroma tree* for both channels. This is especially important for small values of r, since the reduced cost of a shared tree frees up bits for additional or less coarsely quantised data points. In contrast to the first step, we can now also optimise our parameters, known data, and chroma values w.r.t. the final MSE after transformation of the reconstruction back to RGB space. Figure 8.7 shows that the joint chroma mask resulting from this step contains significantly less data than its luma counterpart, but represents colour information accurately nonetheless.

Decompression is very straightforward and strictly sequential (see Fig. 8.8). A PAQ decoding yields access to the parameters and known data. With the trees, we reconstruct the inpainting masks for all channels. We need to inpaint the luma channel first, since we need the structural information of the reconstructed brightness component. This allows us finish the reconstruction by colourising the image with a transformation from YCbCr to RGB space.

File Format. We first write all data sequentially in a binary file. The header contains the image dimensions n_x, n_y, the contrast parameter λ, and the LP factor r, each as a binary number of adequate size. Note that we limit r to the values $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$, and $\frac{1}{16}$ such that we can encode it with 2 bits. The subdivision schemes are stored as a minimal and maximal depth and a binary sequence that indicates the tree structure in-between, just as in the original R-EED. In addition, we also write the quantised luma or chroma values as a sequence of raw binary numbers. Finally, we apply the entropy coder PAQ [178] to the whole file. Fig. 8.9 shows a visualisation of this file structure.

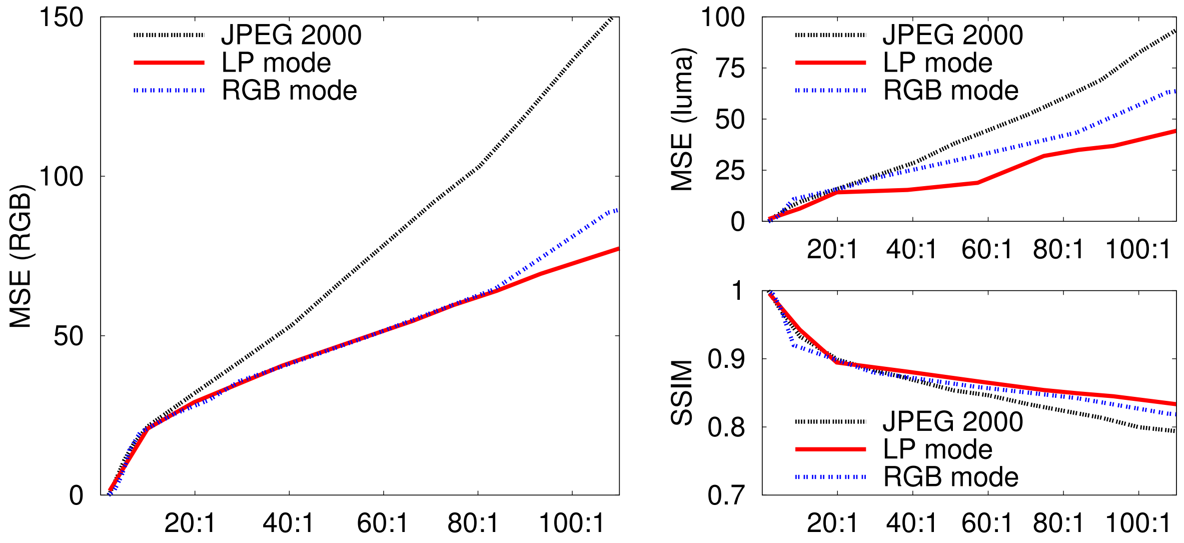


Figure 8.10.: **Error Comparison at Different Compression Ratios for the** 256×256 **Image** ***peppers*****. (a) Left:** Joint MSE in RGB space. Lower is better. **(b) Top right:** MSE in the luma channel. **(c) Bottom right:** SSIM. Higher is better.

Table 8.3.: Comparison of Compression Techniques on the Kodak Database at Ratio $60 : 1$.

Algorithm	*Average Ratio*	*Average MSE*	*Maximum MSE*	*Average SSIM*	*Minimum SSIM*
JPEG	59.40	109.50	341.52	0.83	0.665
JPEG2000	59.37	78.91	278.83	0.84	0.74
LP-EED	59.38	84.95	309.66	0.818	0.714

8.2.5 Compression Experiments

We compare our luma preference codec to the state-of-the-art in PDE-based compression, namely R-EED in RGB space, and the established transform coders JPEG and JPEG2000. For a quantitative evaluation, we consider mainly the mean square error (MSE) over all three channels of RGB space. Since structural information is particularly important for perception, we also use the luma channel MSE and the structural similarity index (SSIM), a perceptually motivated measure for greyscale-converted colour images [289].

In previous chapters, we have conducted experiments only on a small set of standard test images. Here, in addition, we speed up our algorithm such that for the first time, we can perform experiments with PDE-based codecs on the Kodak database [83]. This database is used frequently for the evaluation of compression methods in the industry [2, 108]. In particular, we replace the SOR scheme of R-EED by the FED-approach with coarse-to-fine initialisation from Section 8.2.2 and parallelise it on the GPU. On *peppers*, additional technical modifications to the parameter optimisation reduce the runtime for a full compression from weeks (R-EED) to ≈ 5 hours on an *Intel Xeon CPU W3565@3.20GHz* with an *Nvidia Geforce GTX 460*. A typical compression without brightness optimisation is done in ≈ 7 minutes and the corresponding decompression takes ≈ 0.7 seconds.

Let us first consider the test image *peppers*, which contains only moderate amounts of texture. This is well-suited for our algorithm since it fits the smoothness assumption of our diffusion-based inpainting. Therefore, even R-EED in RGB-mode can outperform JPEG2000 consistently (see Fig. 8.10 (a)) in regards to the overall MSE. However, LP mode offers additional perceived quality with a more accurate luma channel (Fig. 8.10 (b) and (c)). For high compression ratios, it improves both the luma and the chroma MSE compared to R-EED (RGB). Even at a ratio of $110 : 1$ in Fig. 8.11 (a), our codec still produces a good representation of the original image while JPEG2000 reaches its limits. The transform-based coders suffer from block-artefacts at this ratio and the overall error is almost twice as high for JPEG2000. The same behaviour can also be observed for the test image *lena* at low to medium compression ratios (see Fig. 8.11 (b)).

On the natural images of the Kodak database, transform-based coders have the advantage that they can deal better with highly textured images. On images from the database with a medium amount of texture, LP mode outperforms all competitors, while R-EED (RGB) stays behind JPEG and JPEG2000 (see Fig. 8.11 (c)). Even considering the whole database including also highly textured images in Table 8.3, LP mode beats JPEG consistently in regards to the RGB MSE. JPEG has still a slight advantage in the luma channel, but only since it drives chroma subsampling to the extreme. In some cases, very severe colour artefacts are visible. The average quality is only slightly behind JPEG2000, even though texture is not specifically addressed by this codec.

Publication Info

The colourisation and colour compression in this chapter have been published by Peter, Kaufhold, and Weickert [225] in a technical report. This thesis contains additional experiments with different discretisation parameters. Note that the technical report extends on previous publications: The basic ideas of diffusion-based colourisation have been investigated first by Lilli Kaufhold [150] in her Bachelor's thesis under supervision of Joachim Weickert. The compression part of this chapter extends preliminary results presented by Peter and Weickert [219] at the IEEE International Conference on Image Processing 2014.

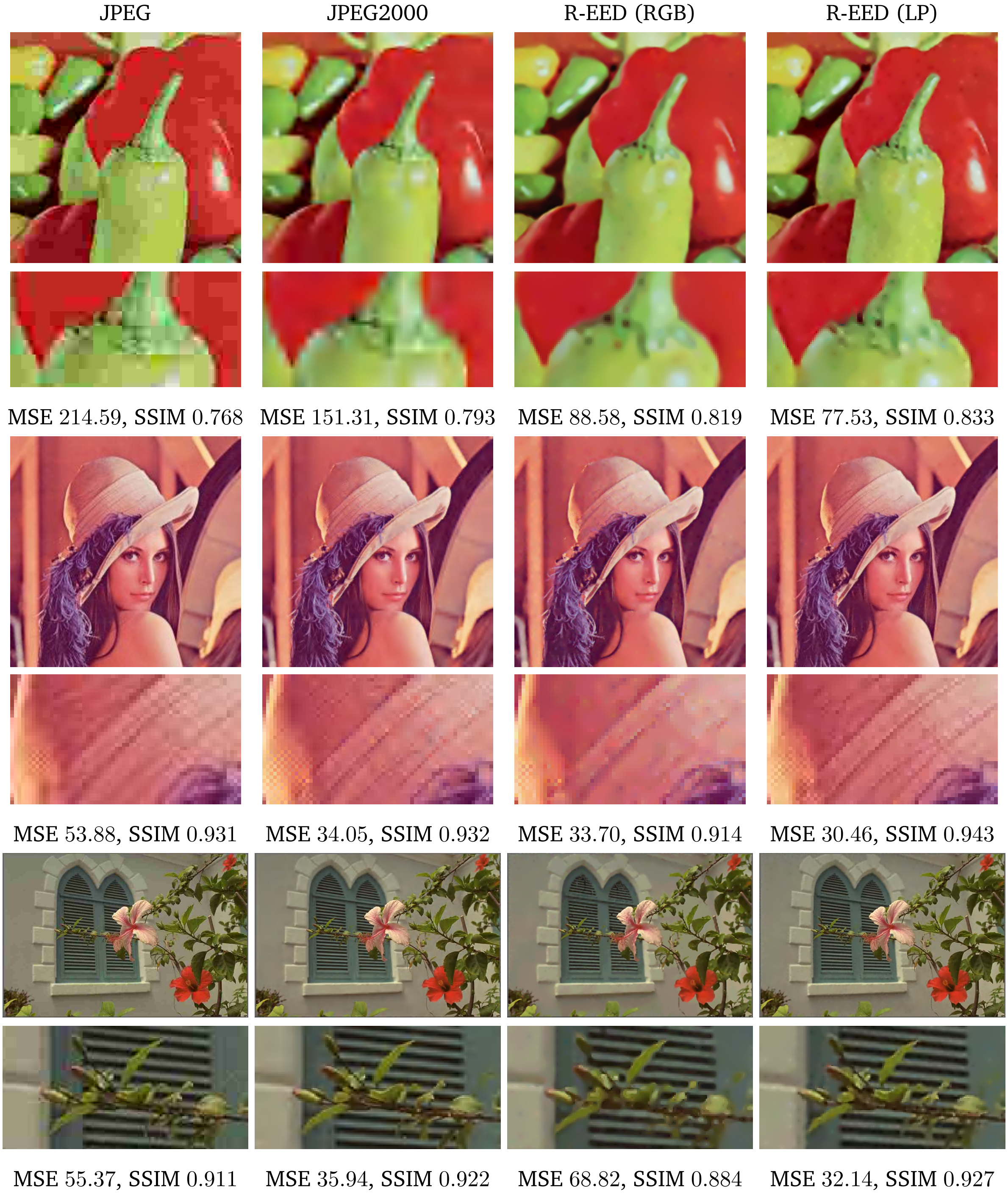


Figure 8.11.: **Compression Results (Overall MSE and SSIM) for *peppers*, *lena*, and *kodim07* from the Kodak Database. (a) Top:** Results for *peppers* (256 × 256 pixels) with compression ratio 110:1. **(a) Middle:** Results for *lena* (256 × 256 pixels) with compression ratio 20:1. **(b) Bottom:** Results for *kodim07* (768 × 512 pixels) with compression ratio 60:1.

8.3 Conclusion

> *There are things known and there are things unknown, and in between are the doors of perception.*
>
> — **Aldous Huxley**
> (Writer)

With this chapter, we have introduced perceptive coding to PDE-based compression for the first time. Our region of interest coding by error weighting shows that optimisation-driven diffusion codecs can easily adapt image quality in different regions without creating overhead. This is an important addition for the use of PDE-based compression in specialised areas like medical imaging.

Moreover, we have presented a new diffusion-based colourisation framework in YCbCr space and have shown that the method of Levin et al. is related to isotropic diffusion inpainting. An evaluation shows that luma-guided anisotropic diffusion outperforms the approach of Levin et al. and can reconstruct colour from very sparse known data. This makes it a valuable tool for compression. Our luma preference mode is the first codec that introduces perceptive coding to PDE-based compression. An evaluation on well-known test images and the Kodak database demonstrates that on pictures with medium amounts of texture, LP mode can beat both PDE- and transform-based competitors. Even for highly textured images, our method outperforms JPEG and comes close to the quality of JPEG2000.

Our contributions open up interesting perspectives for future work: In combinations with optic flow or 3-D diffusion, luma-guided diffusion is promising for video colourisation. Moreover, the new higher-order models could be evaluated in different applications. On the compression side, the performance on highly textured images can be addressed by applying texture synthesis in the luma channel. Hybrid approaches with non-local and PDE-based inpainting have already shown promise [220] in this regard. Overall, our contributions demonstrate that there is still a lot of unlocked potential for new applications in diffusion-based image processing. In particular, with perceptive coding of colour images, we provide another important step towards reaching the same sophisticated level of engineering as transform-based coders.

Conclusions and Outlook 9

> *The wheel is come full circle, I am here.*
>
> — **William Shakespeare**
> (Dramatist)

In the beginning of this thesis, we set out to analyse the status quo of PDE-based compression and develop a deeper understanding of its success and limitations. Furthermore, we proposed different categories for successful codecs based on the requirements for JPEG by Pennebaker and Mitchell [215] as a guideline for new features in PDE-based compression. Throughout the thesis, we have addressed four of these categories: quality, consistency, immediacy and perceptive coding. In the following, we discuss in how far our findings in each area contribute to the field and what they imply for future research.

General Observations

From a modelling perspective, we have found close, systematic connections between tensor-driven diffusion processes and natural image priors in Chapter 4. On the one hand, this allows us to integrate eight existing diffusion models into a unifying, statistically-motivated framework and provide a new justification for the success of anisotropic diffusion in PDE-based compression. On the other hand, it is particularly intriguing that the historical evolution of PDE-based inpainting was originally a task-driven development process in which experts developed increasingly better smoothness assumptions. From a statistical perspective, however, this evolution can be interpreted as a gradual improvement of approximation quality towards a structure tensor prior that measures anisotropy in natural images.

The aforementioned findings concentrate on the general capabilities of individual PDEs in image processing. For the specific purpose of compression, however, a lot of additional factors determine the overall performance: the selection and storage efficiency of known data, quantisation, and entropy coding can affect the trade-off between file size and reconstruction quality. In Chapter 5, we have proposed two new codecs: the first one uses optimal control to optimise pixel-accurate inpainting masks, a technique that no codec uses so far. Our second codec restricts point positions to a regular adaptive grid that can be efficiently stored by a binary tree representation, sacrificing reconstruction quality for smaller coding cost. In contrast to established techniques like R-EED, we do not use a heuristic to determine the tree, but instead employ a probabilistic densification technique.

Both codecs allow an evaluation with the most popular reconstruction algorithms in compression: harmonic, biharmonic, and edge-enhancing diffusion (EED). Each inpainting technique is combined with a tonal optimisation step as postprocessing. This setup allows a thorough analysis of the interplay between optimisation techniques for inpainting and compression steps. Coarse quantisation that allows more efficient entropy coding deteriorates compression quality. However, this negative effect can be partially compensated by tonal optimisation. Interestingly, the higher-order biharmonic inpainting is more sensitive to quantisation than homogeneous diffusion and EED.

While these experiments confirm the overall superiority of EED and thereby motivate the use of EED in the rest of the thesis, the different sensitivity of inpainting operators to compression also leads to surprising results. In other studies that consider the inpainting capabilities without actually storing data [59, 125], biharmonic inpainting beats the simpler harmonic operator. In practice, however, it turns out that quantisation and entropy encoding actually favour homogeneous inpainting in certain situations. These results show that it is *vital to consider PDEs in the actual context of a codec*, since comparative studies that ignore compression steps can yield misleading results.

Practical Viability

Building on the general insights from the previous sections, we addressed the four areas that we have identified as limitations of the current generation of PDE-based codecs in Chapter 1.

Consistency. Since texture does not fit to the smoothness assumptions that are enforced in the inpainting domain by diffusion operators, PDE-based compression struggles with images with a large amount of small-scale texture. To this end, we have proposed a straightforward block decomposition scheme in Chapter 6 that allows to augment diffusion-based reconstructions with additional texture synthesis methods and produces minimal overhead. We have evaluated the capabilities of this hybrid framework in combination with the patch-based inpainting approach for sparse known data by Facciolo et al. [90]. This leads to significant improvements for images with regular texture and allows to surpass JPEG2000. Interestingly, a surprising amount of robustness under texture data also comes from another codec that we introduced in Chapter 8.2, which was originally intended for another purpose. Our colour compression codec allows us to store a lot more of the texture that cannot be reconstructed by PDE-based inpainting and allows us to surpass JPEG and come close to JPEG2000 without resorting to patch-based inpainting at all on the Kodak image database [83].

Immediacy. The second big challenge for PDE-based compression is runtime. The powerful inpainting capabilities of EED are bought at the price of more challenging numerics and higher runtime in comparison to simple linear inpainting. For video coding, we have demonstrated that real-time performance is possible on customer hardware with powerful solvers that are simultaneously simple to implement. The black box fast explicit diffusion (FED) solver by Weickert et al. [298] offers a very efficient and straightforward way to implement EED inpainting. In particular, it is also well-suited for parallelisation on customer class GPUs. In combination with a smart initialisation system that reuses as much of the previous frame to speed up the convergence of the next inpainting, we achieve real-time performance. With a sophisticated caching system, the 640×480 movie *Nosferatu* can be played back at 25 frames per second. In particular, we reach a compression ratio of 20:1 without making noticeable compromises in regards to reconstruction quality in comparison to full regular frame-by-frame decompression without time constraints. Before the thesis, this was only possible for videos with significantly lower resolution and by sacrificing quality [22, 156].

Perceptive Coding. In Chapter 8 we have combined many of the findings from the rest of the thesis. Since the combination of binary subdivision trees and EED is qualitatively still the most advanced method, we propose two new features for tree-based compression: region of interest coding and colour compression.

Our ROI extension is backwards compatible with R-EED [252], but can also be integrated in our new codecs from this thesis. It is simultaneously simple to implement, powerful, and flexible. The user defines a pixel-accurate weighting of the error measure that is used for spatial and tonal optimisation. This creates no additional overhead in the compressed file and the subdivision scheme provides an automatic adaptation of the regular grid to ROIs of arbitrary shape.

The novel colour compression codec from Chapter 8.2 also falls in the category of perceptive coding since it exploits the fact that the human visual system values structure higher than colour. It is based on a new colourisation technique, luma-guided inpainting. Here, a decomposition into brightness and colour information enables us to reuse structural information from the brightness or luma channel to get a good reconstruction in the chroma channels with very few known data. We first show the relationship of this method to the approach of Levin et al. [164]. The anisotropy in our approach leads to a large qualitative advantage that we can exploit in our novel codec. By allowing the brightness channel a large file size budget, structural information can be stored accurately. Due to the luma-guided inpainting, also the chroma channels benefit from this information. This allows us to outperform both the state-of-the art in PDE-based image compression, R-EED, as well as JPEG2000, by a large margin on images with medium and low amounts of texture.

Furthermore, our experiences with real-time processing in video compression allow us to evaluate PDE-based codecs under more realistic conditions than before. For the first time, we are able to conduct experiments on a well-recognised database of natural images [83]. Since this database contains a lot of images with high amounts of texture, it is ill-posed for our codec. Nevertheless, we are able to beat JPEG consistently by a large margin and even come close to JPEG2000.

Summary. In total, we have introduced important new features, namely region of interest and colour coding. Furthermore, we have proposed ways to improve robustness for textured image content and improved runtime performance. This also allowed us to assess the current state of PDE-based compression for the first time under realistic conditions on a natural image database, showing that it vastly outperforms JPEG and comes close to JPEG2000, even when heavy texture is involved. All of these new contributions are compatible with important existing features for PDE-based compression such as progressive modes [251] or steganography [182].

9.1 Future Work

> ” *Prediction is very difficult,*
> *especially if it's about the future.*
>
> — **Niels Bohr**
> (Physicist, Winner of the Nobel Prize)

While we have found answers to many important questions from multiple areas of PDE-based compression, just as many new questions arise. In the following we sketch ideas for additional theoretical and practical research topics that go beyond our contributions.

Theoretical Perspectives

There are many opportunities to extend on the relationship between statistical priors and diffusion. So far, we have used statistics mainly to justify existing models. In this regard, the search for a variational formulation of EED is still not resolved. So far, it has neither been proven nor disproven if a corresponding energy functional exists. However, our analysis of directional derivative priors in Chapter 4.8 gives another hint that EED is closely related to the statistics of natural images. In future research, we want to investigate these relations more closely.

Beyond the search for correspondences to prior work, it is also attractive to motivate new and potentially more powerful diffusion models from statistics: The fact that probabilistic models like Field-of-Experts (FoE) [238] achieve better results with learned filters than with fixed features suggests that there might be also better corresponding diffusion models. Many of the learned features still resemble derivative filters or show some symmetries. Thereby, it could be fruitful to investigate if there is a consistent underlying pattern to the learned filters that implies a new class of diffusion or diffusion-reaction filters.

Moreover, our experiments in Chapter 4.8 suggest that the learned models are not as well-suited for inpainting as for denoising, since they do not incorporate different distributions of known data and only incorporate localised statistics. Therefore, it seems viable to investigate new, specifically designed inpainting priors that investigate more global relationships between different MRF random variables.

In a similar direction, texture priors could be a valuable research topic. Exemplar-based inpainting methods also exploit global interactions between image patches to reconstruct texture. Recently, a theory for non-local diffusion processes has been established by Cárdenas et al. [46]. Possibly, the prior-framework could also be extended to contain such non-local approaches.

Furthermore, little research on PDE-based compression has been conducted from the point of view of information theory. Only Scheer [247] tried to deviate from standard entropy coders, albeit with limited success. However, the current use of entropy coders like PAQ [178] discards a lot of valuable information. Instead of simply encoding a sequence of grey values and ignoring the spatial relationships of mask locations, new file type models could be introduced to entropy coders that incorporate this information.

Towards Practical Viability

For some specific classes of images like depth-maps and cartoon images where the smoothness assumption is fulfilled, diffusion-based codecs are already viable. However, for general purpose encoding, it is clear that even after the progress made in this thesis, texture and runtime remain the two most pressing issues that have to be addressed before a wide-spread use of these codecs is possible.

Our block-based approach for texture encoding is successful, but can still be improved. Currently, it conducts spatial optimisation only in regards to PDE. The exemplar-based inpainting only comes into play in postprocessing. A refined method should treat both reconstruction approaches equally. Beyond block-based encoding, a unifying inpainting approach would be even more promising. Such a method would locally adapt to the image structure and decide if structure propagation or texture synthesis is more efficient in a given image area. Recent progress on non-local diffusion models [46] could provide the tools for such an endeavour. Furthermore, learned multi-stage diffusion processes [58] seem to be able to reproduce texture from sparse data remarkably well [318].

The aforementioned multi-stage models could also help in regards to runtime, since they only require a limited number of iterations by definition. Furthermore, the results from Chapter 5 show that especially for low compression ratios, linear models seem to be quite promising. For these models, there are strong indications that very efficient implementations with runtimes comparable to JPEG and JPEG2000 are possible [128].This could also be exploited in combination with the colour compression from Chapter 8.2 to create a completely linear compression pipeline: The brightness channel could be compressed with a low compression ratio using linear homogeneous diffusion. Based on the accurate structural information from this compression, a linear anisotropic colourisation approach can then be applied to reconstruct colour information. This would help to reduce the bottleneck of tonal optimisation which currently makes up most of the runtime of compression.

In regard to brightness optimisation, it could also be viable to explore localisation strategies based on the influence regions of the mask points. Since localised diffusion is already employed successfully for subdivision trees in R-EED [252] and this thesis, it would be consequent to also try this not only for spatial, but also for tonal optimisation.

Ultimately, PDE-based compression has to aim for synchronous codecs where compression and decompression are both possible in real-time. Especially for widespread use in browsers this is vital. Of course, standardisation and patent-free components are also important. Practitioners like Google [108] and Mozilla [2] both invest heavily into replacements for JPEG in webbrowers and consider the aforementioned criteria just as important as quality [1].

In order to go towards standardisation, it would help to establish a common platform for PDE-based codecs that unifies the engineering aspects. PDE-based compression is largely model-driven and uses ready-made entropy coding techniques. Interoperability could be improved and the results made available to a broader audience by standardising. Furthermore, such a framework would allow to design a standardised decoder that ensures backwards-compatibility while encoders can be improved over the years. This concept was very successful for JPEG. In Appendix I we provide a rough sketch of how such a framework could look like. In particular, it would also be helpful to propose benchmarking tools for PDE-based compression together with such a framework. As Chapter 5 shows, comparative studies in PDE-based compression are currently not standardised and can easily lead to wrong conclusions if not all aspects of compression are properly considered.

Extensions

Beyond general purpose encoding, chances of pure diffusion lie in image areas where the smoothness assumption is fulfilled or structural data is cheap to store. For example, there are dedicated transform-based codecs for remote sensing, which includes hyperspectral data (see Serra-Sagristà and Aulí-Llinàs [256] for an overview). Here, a large number of channels with highly correlated edges have to be compressed. Since luma-guided inpainting from Chapter 8.2 seems to exploit the structural correlation in multi-channel images well, hyperspectral imaging could be a good specialised application for PDE-based inpainting. In particular, it would be interesting how PDE-based encoding would impact subsequent practical applications (such as classification of the compressed images) compared to transform-based coders [102]. Furthermore, tensor data from medical imaging could also be a content type that might benefit from PDE-based compression.

Obviously, video compression is another big area that has seen only small amounts of attention in the PDE-based compression community. A particularly interesting starting point seems to be cartoon videos. Here, linear diffusion approaches [184] are very successful on single images quality wise. Simultaneously, achieving real-time performance is easier than for non-linear diffusion models. So far, the possibility of using redundancies between different frames for PDE-based encoding remains virtually unexplored. This could be either achieved by prediction as in current transform-based coders, or by applying 3-D diffusion [218].

Furthermore, current PDE-based compression methods rely primarily on storing selected pixels and thus creating a sparse representation of the image in the spatial domain. However, diffusion-based reconstruction is not necessarily limited to this kind of known data. Only few methods try to prescribe other kinds of data, namely gradients [39, 254]. One could also consider to specify diffusivities or whole diffusion tensors explicitly at key points in the image. Such ideas could be interesting whenever the trade-off between storing diffusion tensors verbatim and storing them indirectly via the configuration of several pixels is in favour of the tensor. Closest to this approach is the reproduction of geometric shapes by Huth [138] and Steil [265] that relies on carefully chosen quadrupoles or nonapoles.

Another very important future field for PDE-based compression could be psycho-visual encoding. In video coding, an increasing number of codecs emerge that do not rely purely on classic quantitative metrics like the MSE. Instead, they explicitly consider highly textured regions as "perceptually irrelevant" [203, p. 2] and try to substitute them with synthesised textures that are only visually similar. However, such visual similarity is very difficult to detect and current methods still struggle with finding adequate metrics. A good overview over different methods is given by Ndjiki-Nya et al. [203].

9.2 Closing Words

> ”*Despair is the conclusion of fools.*
>
> — **Benjamin Disraeli**
> (Politician and Writer)

Since the dawn of PDE-based compression, critics had it always easy to dismiss this new type of codecs and even now some of the initial arguments are still used against it: The texture problem is not completely solved and runtime is not on par with JPEG and JPEG2000.

In most cases, it requires little effort to provide a pessimistic view on competitors to established methods. However, many of the initial arguments have been proven wrong: The argument that JPEG could not be beaten was countered already by Galić et al. [99]. The same assumption on the general superiority of JPEG 2000 was set aside by Schmaltz et al. [252] for grey value images and in Chapter 8.2 we did the same for colour images. Our real-time video compression from Chapter 7 shows that the scepticism regarding the performance of anisotropic diffusion can be overcome with tenacity and efficient use of available information. All of these examples indicate that even the small current compression community can overcome many challenging issues that remain.

The contributions of this thesis and the outlook in the previous section convey a current trend: PDE-based compression is gaining momentum. An honest assessment of the current situation has to admit that the journey towards practical viability is still a difficult one. However, promising results indicate that the most pressing issues are either getting resolved or can be addressed in the future. As more researchers show their interest in the topic, the future of PDE-based compression might very well be a bright one.

Bibliography

A

> *An apt quotation is like a lamp*
> *which flings its light over the whole sentence.*
>
> — **Letitia E. Landon**
> (Writer)

[1] J. Aas: *Studying Lossy Image Compression Efficiency*, 2013, Available at `https://blog.mozilla.org/research/2013/10/17/studying-lossy-image-compression-efficiency/`, visited on October 4, 2015 (cited on p. 192).

[2] J. Aas, G. Maxwell, J. Muizelaar, and T. Terriberry: *Lossy Compressed Image Formats Study*, 2013, Available at `http://people.mozilla.org/~josh/lossy_compressed_image_study_october_2013/`, visited on October 4, 2015 (cited on pp. 182, 192).

[3] M. Abomhara, O. O. Khalifa, O. Zakaria, A. A. Zaidan, B. B. Zaidan, and A. Rame: „Video compression techniques: An overview“, *Journal of Applied Sciences*, Vol. 10, No. 16, 1834–1840, June 2010 (cited on p. 53).

[4] T. Acar and M. Gökmen: „Image coding using weak membrane model of images“, In A. K. Katsaggelos (Ed.): *Visual Communications and Image Processing*, Proceedings of SPIE, Vol. 2308, 1221–1230, SPIE Press, Bellingham, 1994 (cited on p. 57).

[5] J. Aczél and Z. Daróczy: *On Measures of Information and their Characterizations*, Mathematics in Science and Engineering, Vol. 115, Academic Press, New York, 1975 (cited on p. 47).

[6] A. Agrawal, R. Raskar, and R. Chellappa: „What is the range of surface reconstructions from a gradient field?“, In H. Bischof, A. Leonardis, and A. Pinz (Eds.): *Computer Vision – ECCV 2006, Part I*, Lecture Notes in Computer Science, Vol. 3951, 578–591, Springer, Berlin, 2006 (cited on p. 69).

[7] M. Aharon, M. Elad, and A. Bruckstein: „K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation“, *IEEE Transactions on Signal Processing*, Vol. 54, No. 11, 4311–4322, November 2006 (cited on p. 131).

[8] F. Alter, S. Durand, and J. Froment: „Adapted total variation for artifact free decompression of JPEG images“, *Journal of Mathematical Imaging and Vision*, Vol. 23, No. 2, 199–211, September 2005 (cited on p. 53).

[9] H. A. Aly and E. Dubois: „Image up-sampling using total-variation regularization with a new observation model“, *IEEE Transactions on Image Processing*, Vol. 14, No. 10, 1647–1659, October 2005 (cited on p. 35).

[10] P. Arias, G. Facciolo, V. Caselles, and G. Sapiro: „A variational framework for exemplar-based image inpainting“, *International Journal of Computer Vision*, Vol. 93, No. 3, 319–347, July 2011 (cited on p. 130).

[11] G. Aronsson: „Extension of functions satisfying Lipschitz conditions“, *Arkiv för Matematik*, Vol. 6, No. 28, 551–561, June 1967 (cited on pp. 38, 61).

[12] I. Ashdown: „PCX graphics“, *C Users Journal*, Vol. 9, No. 8, 89–86, August 1991 (cited on p. 49).

[13] V. Aurich and U. Daub: „Bilddatenkompression mit geplanten Verlusten und hoher Rate“, In B. Jähne, P. Geißler, H. Haußecker, and F. Hering (Eds.): *Mustererkennung 1996: Tagungsband des 18. DAGM-Symposiums*, Informatik aktuell, 138–146, Springer, Berlin, 1996 (cited on p. 61).

[14] E. Bae and J. Weickert: „Partial differential equations for interpolation and compression of surfaces“, In M. Daehlen, M. Floater, T. Lyche, J.-L. Merrien, K. Mørken, and L. L. Schumaker (Eds.): *Mathematical Methods for Curves and Surfaces*, Lecture Notes in Computer Science, Vol. 5862, 1–14, Springer, Berlin, 2010 (cited on p. 60).

[15] A. Bain: *Production and Regulation of Electric Currents, Electric Timepieces and Electric Printing and Signal Telegraphs*, Patent GB 9,745, 1743 (cited on p. 46).

[16] C. Ballester, V. Caselles, J. Verdera, M. Bertalmío, and G. Sapiro: „A variational model for filling-in gray level and color images“, *Proc. 18th IEEE International Conference on Computer Vision*, Vol. 1, 10–16, Vancouver, Canada, July 2001 (cited on p. 36).

[17] C. Ballester, M. Bertalmío, V. Caselles, G. Sapiro, and J. Verdera: „Filling-in by joint interpolation of vector fields and gray levels“, *IEEE Transactions on Image Processing*, Vol. 10, No. 8, 1200–1211, August 2001 (cited on p. 36).

[18] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst: *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2nd Edition, SIAM, Philadelphia, 1994 (cited on p. 44).

[19] J. Bartrina-Rapesta, J. Serra-Sagristà, and F. Aulí-Llinàs: „JPEG2000 ROI coding through component priority for digital mammography“, *Computer Vision and Image Understanding*, Vol. 115, No. 1, 59–68, January 2011 (cited on p. 164).

[20] 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)*, Vol. 11, No. 2, 92–100, January 2010 (cited on pp. 55, 57).

[21] S. Battiato, G. Gallo, and F. Stanco: „Smart interpolation by anisotropic diffusion“, *Proc. Twelvth International Conference on Image Analysis and Processing*, 572–577, Mantova, Italy, September 2003 (cited on p. 35).

[22] K. Baum: „GPGPU Supported Diffusion-Based Naive Video Compression“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on pp. 6, 59, 146, 189).

[23] F. Beekman, E. Slijpen, and W. Niessen: „Supervised diffusion parameter selection for filtering SPECT brain images“, In B. ter Haar Romeny, L. Florack, J. Koenderink, and M. Viergever (Eds.): *Scale-Space Theory in Computer Vision*, Lecture Notes in Computational Science and Engineering, Vol. 1252, 164–175, Springer, Berlin, 1997 (cited on p. 75).

[24] A. Belahmidi and F. Guichard: „A partial differential equation approach to image zoom“, *Proc. 2004 IEEE International Conference on Image Processing*, Vol. 1, 649–652, Singapore, October 2004 (cited on p. 35).

[25] Z. Belhachmi, D. Bucur, B. Burgeth, and J. Weickert: „How to choose interpolation data in images“, *SIAM Journal on Applied Mathematics*, Vol. 70, No. 1, 333–352, June 2009 (cited on pp. 56, 61, 64, 96, 101–103).

[26] M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester: „Image inpainting“, *Proc. SIGGRAPH 2000*, 417–424, New Orleans, LI, July 2000 (cited on pp. 35, 36).

[27] M. Bertalmío, L. Vese, G. Sapiro, and S. Osher: „Simultaneous structure and texture image inpainting“, *IEEE Transactions on Image Processing*, Vol. 12, No. 8, 882–889, August 2003 (cited on p. 130).

[28] J. Besag: „Spatial interaction and the statistical analysis of lattice systems“, *Journal of the Royal Statistical Society B*, Vol. 36, No. 2, 192–236, March 1974 (cited on p. 76).

[29] F. Bornemann and P. Deuflhard: „The cascadic multigrid method for elliptic problems“, *Numerische Mathematik*, Vol. 75, No. 2, 135–152, December 1996 (cited on p. 149).

[30] F. Bornemann and T. März: „Fast image inpainting based on coherence transport“, *Journal of Mathematical Imaging and Vision*, Vol. 28, No. 3, 259–278, July 2007 (cited on p. 38).

[31] 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*, Vol. 7, No. 3, 410–425, July 1998 (cited on pp. 50, 111, 116).

[32] S. Bougleux, G. Peyré, and L. Cohen: „Image compression with anisotropic triangulations“, *Proc. 12th International Conference on Computer Vision*, 2343–2348, Kyoto, Japan, September 2009 (cited on p. 62).

[33] P. Bourdon, B. Augereau, C. Olivier, and C. Chatellier: „MPEG-4 compression artifacts removal on color video sequences using 3D nonlinear diffusion“, *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3, 729–732, Montreal, Canada, May 2004 (cited on p. 53).

[34] A. Bourquard and M. Unser: „Anisotropic interpolation of sparse generalized image samples“, *IEEE Transactions on Image Processing*, Vol. 22, No. 2, 459–472, February 2013 (cited on p. 38).

[35] T. Boutell: *PNG (Portable Network Graphics) Specification Version 1.0*, RFC 2083, Status: Informational, January 1997 (cited on p. 49).

[36] C. J. van den Branden Lambrecht (Ed.): *Vision Models and Applications to Image and Video Processing*, Kluwer, Boston, 2001 (cited on p. 178).

[37] A. Brandt: „Multi-level adaptive solutions to boundary-value problems“, *Mathematics of Computation*, Vol. 31, No. 138, 333–390, April 1977 (cited on p. 149).

[38] K. Bredies and M. Holler: „A total variation-based JPEG decompression model“, *SIAM Journal on Applied Mathematics*, Vol. 5, No. 1, 366–393, March 2012 (cited on p. 53).

[39] E.-M. Brinkmann, M. Burger, and J. Grah: „Regularization with sparse vector fields: From image compression to TV-type reconstruction“, In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker (Eds.): *Scale-Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 9087, 191–202, Springer, Berlin, 2015 (cited on pp. 61, 193).

[40] T. Brox and D. Cremers: „On the statistical interpretation of the piecewise smooth Mumford-Shah functional“, In F. Sgallari, A. Murli, and N. Paragios (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 4485, 203–213, Springer, Berlin, 2007 (cited on p. 75).

[41] D. Brunet, E. Vrscay, and Z. Wang: „On the mathematical properties of the structural similarity index“, *IEEE Transactions on Image Processing*, No. 4, 1488–1499, April 2012 (cited on p. 24).

[42] M. Burrows and D. Wheeler: *A block-sorting lossless data compression algorithm*, Technical Report No. 124, Digital Systems Research Center, Palo Alto, CA, May 1994 (cited on p. 49).

[43] J. Canny: „A computational approach to edge detection“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, 679–698, November 1986 (cited on p. 57).

[44] F. Cao, Y. Gousseau, S. Masnou, and P. Pérez: „Geometrically guided exemplar-based inpainting“, *SIAM Journal on Imaging Sciences*, Vol. 4, No. 4, 1143–1179, December 2011 (cited on p. 130).

[45] J. Capon: „A probabilistic model for run-length coding of pictures“, *IRE Transactions on Information Theory*, Vol. 5, No. 4, 157–163, December 1959 (cited on p. 49).

[46] G. M. Cárdenas, J. Weickert, and S. Schäffer: „A linear scale-space theory for continuous nonlocal evolutions“, In J.-F. Aujol, M. Nikolova, and N. Papadakis (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 9087, 103–114, Springer, Berlin, 2015 (cited on p. 191).

[47] S. Carlsson: „Sketch based coding of grey level images“, *Signal Processing*, Vol. 15, No. 1, 57–83, July 1988 (cited on p. 57).

[48] V. Caselles, J.-M. Morel, and C. Sbert: „An axiomatic approach to image interpolation“, *IEEE Transactions on Image Processing*, Vol. 7, No. 3, 376–386, March 1998 (cited on p. 36).

[49] K. R. Castleman: *Digital Image Processing*, Prentice Hall, Englewood Cliffs, 1996 (cited on p. 16).

[50] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll: „Image selective smoothing and edge detection by nonlinear diffusion“, *SIAM Journal on Numerical Analysis*, Vol. 29, No. 1, 182–193, February 1992 (cited on p. 32).

[51] T. F. Chan and J. Shen: „Non-texture inpainting by curvature-driven diffusions (CDD)“, *Journal of Visual Communication and Image Representation*, Vol. 12, No. 4, 436–449, December 2001 (cited on p. 36).

[52] T. F. Chan and J. Shen: „Variational image inpainting“, *Communications on Pure and Applied Mathematics*, Vol. 58, No. 5, 579–619, May 2005 (cited on p. 93).

[53] T. F. Chan and H. M. Zhou: „Total variation improved wavelet thresholding in image compression“, *Proc. Seventh IEEE International Conference on Image Processing*, Vol. 2, 391–394, Vancouver, Canada, September 2000 (cited on p. 57).

[54] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud: „Two deterministic half-quadratic regularization algorithms for computed imaging“, *Proc. First IEEE International Conference on Image Processing*, Vol. 2, 168–172, Austin, TX, November 1994 (cited on pp. 34, 72, 89, 134).

[55] D. Charlap: „The BMP file format“, *Dr. Dobb's Journal – Software Tools for the Professional Programmer*, Vol. 20, No. 3, 44–51, March 1995 (cited on p. 49).

[56] J. Chen, F. Ye, J. Di, C. Liu, and A. Men: „Depth map compression via edge-based inpainting“, *Proc. 29th Picture Coding Symposium*, 57–60, Kraków, Poland, May 2012 (cited on p. 62).

[57] S. Chen: „Image reconstruction from zero-crossings“, *Proc. 10th International Joint Conference on Artificial Intelligence*, Vol. 2, 742–744, Milan, Italy, August 1987 (cited on p. 57).

[58] Y. Chen, W. Yu, and T. Pock: „On learning optimized reaction diffusion processes for effective image restoration“, *Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition*, 5261–5269, Boston, MA, June 2015 (cited on pp. 75, 130, 191).

[59] Y. Chen, R. Ranftl, and T. Pock: „A bi-level view of inpainting-based image compression“, *Proc. 19th Computer Vision Winter Workshop*, 19–26, Křtiny, Czech Republic, February 2014 (cited on pp. 38, 56, 64, 67, 96, 101–105, 109, 115, 122, 188).

[60] L. Cheng and S. Vishwanathan: „Learning to compress images and videos“, *Proc. 24th International Conference on Machine Learning*, 161–168, Corvallis, OR, June 2007 (cited on p. 169).

[61] Cisco Systems, Inc.: *Cisco Visual Networking Index: Forecast and Methodology, 2013 – 2018*, White Paper, San Jose, CA, 2014 (cited on p. 2).

[62] J. Cleary and I. Witten: „Data compression using adaptive coding and partial string matching“, *IEEE Transactions on Communications*, Vol. 32, No. 4, 396–402, April 1984 (cited on p. 48).

[63] CompuServe Incorporated: *Graphics Interchange Format: A Standard Defining a Mechanism for the Storage and Transmission of Raster-based Graphics Information*, 1987, Available at `http://www.w3.org/Graphics/GIF/spec-gif87.txt`, visited on December 13, 2015 (cited on p. 49).

[64] G.-H. Cottet: „Diffusion approximation on neural networks and applications for image processing", *Proc. Sixth European Conference on Mathematics in Industry*, 3–9, Limerick, Ireland, August 1992 (cited on p. 69).

[65] G.-H. Cottet and M. El Ayyadi: „A Volterra type model for image processing", *IEEE Transactions on Image Processing*, Vol. 7, No. 3, 292–303, March 1998 (cited on p. 40).

[66] J. Crank: *The Mathematics of Diffusion*, 2nd Edition, Oxford University Press, Oxford, 1975 (cited on p. 28).

[67] A. Criminisi, P. Pérez, and K. Toyama: „Region filling and object removal by exemplar-based image inpainting", *IEEE Transactions on Image Processing*, Vol. 13, No. 9, 1200–1212, September 2004 (cited on p. 130).

[68] L. D. Crocker: „PNG: The portable network graphic format", *Dr. Dobb's Journal – Software Tools for the Professional Programmer*, Vol. 20, No. 7, 36–45, July 1995 (cited on p. 49).

[69] S. R. Curtis, A. V. Oppenheim, and J. S. Lim: „Reconstruction of two-dimensional signals from threshold crossings", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 10, 1057–1060, Tampa, FL, April 1985 (cited on p. 57).

[70] E. D'Angelo, L. Jacques, A. Alahi, and P. Vandergheynst: „From bits to images: Inversion of local binary descriptors", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 5, 874–887, May 2014 (cited on p. 59).

[71] L. Demaret, N. Dyn, and A. Iske: „Image compression by linear splines over adaptive triangulations", *Signal Processing*, Vol. 86, No. 7, 1604–1616, July 2006 (cited on p. 62).

[72] L. Demaret and A. Iske: „Advances in digital image compression by adaptive thinning", *Annals of the Marie Curie Fellowship Association*, Vol. 3, 105–109, February 2004 (cited on p. 62).

[73] U. Y. Desai, M. M. Mizuki, I. Masaki, and B. K. P. Horn: *Edge and Mean Based Image Compression*, Technical Report No. 1584 (A.I. Memo), Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, MA, November 1996 (cited on p. 57).

[74] P. Deutsch: *DEFLATE Compressed Data Format Specification version 1.3*, RFC 1951, Status: Informational, January 1996 (cited on p. 49).

[75] S. Di Zenzo: „A note on the gradient of a multi-image", *Computer Vision, Graphics and Image Processing*, Vol. 33, No. 1, 116–125, January 1986 (cited on pp. 72, 170).

[76] R. Distasi, M. Nappi, and S. Vitulano: „Image compression by B-tree triangular coding", *IEEE Transactions on Communications*, Vol. 45, No. 9, 1095–1100, September 1997 (cited on pp. 54, 61).

[77] D. Doshkov, P. Ndjiki-Nya, H. Lakshman, M. Koppel, and T. Wiegand: „Towards efficient intra prediction based on image inpainting methods“, *Proc. Picture Coding Symposium*, 470–473, Nagoya, Japan, December 2010 (cited on pp. 60, 146).

[78] R. Dosselmann and X. D. Yang: „A comprehensive assessment of the structural similarity index“, *Signal, Image and Video Processing*, Vol. 5, No. 1, 81–91, March 2011 (cited on pp. 21, 24).

[79] L. Dron: „The multiscale veto model: A two-stage analog network for edge detection and image reconstruction“, *International Journal of Computer Vision*, Vol. 11, No. 1, 45–61, August 1993 (cited on p. 57).

[80] I. Drori, D. Cohen-Or, and H. Yeshurun: „Fragment-based image completion“, *ACM Transactions on Graphics*, Vol. 22, No. 3, 303–312, July 2003 (cited on p. 130).

[81] J. Duchon: „Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces“, *RAIRO Analyse Numérique*, Vol. 10, No. 3, 5–12, December 1976 (cited on p. 38).

[82] S. Durand and M. Nikolova: „Restoration of wavelet coefficients by minimizing a specially designed objective function“, *Proc. Second IEEE Workshop on Geometric and Level Set Methods in Computer Vision*, 145–152, Nice, France, October 2003 (cited on p. 53).

[83] Eastman Kodak Company: *Kodak True Color Image Suite*, 1999, Available at `http://r0k.us/graphics/kodak/`, visited on October 4, 2015 (cited on pp. 168, 182, 188, 189).

[84] A. A. Efros and T. K. Leung: „Texture synthesis by non-parametric sampling“, *Proc. Seventh IEEE International Conference on Computer Vision*, Vol. 2, 1033–1038, Corfu, Greece, September 1999 (cited on pp. 129–131).

[85] M. Elad, J.-L. Starck, P. Querre, and D. L. Donoho: „Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)“, *Journal of Applied and Computational Harmonic Analysis*, Vol. 19, No. 3, 340–358, November 2005 (cited on p. 131).

[86] J. H. Elder: „Are edges incomplete?“, *International Journal of Computer Vision*, Vol. 34, No. 2/3, 97–122, 1999 (cited on p. 57).

[87] F. W. Murnau (Director), A. Grau, and E. Dieckmann (Producers): *Nosferatu, eine Symphonie des Grauens*, Motion Picture, Restored Version courtesy of the Friedrich-Wilhelm-Murnau foundation, Wiesbaden, 1922 (cited on pp. vi, viii, 145, 148, 159).

[88] Fabrice Bellard: *BPG Specification*, 2014, Available at `http://bellard.org/bpg/bpg_spec.txt`, visited on October 4, 2015 (cited on pp. 2, 53).

[89] G. Facciolo, F. Lecumberry, A. Almansa, A. Pardo, V. Caselles, and B. Rougé: „Constrained anisotropic diffusion and some applications“, *Proc. 2006 British Machine Vision Conference*, Vol. 3, 1049–1058, Edinburgh, Scotland, September 2006 (cited on p. 61).

[90] 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. Schmidt (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 5681, 331–344, Springer, Berlin, 2009 (cited on pp. 8, 129–132, 137, 140, 144, 188, 247).

[91] R. M. Fano: *The Transmission of Information*, Technical Report No. 65, Massachusetts Institute of Technology, Research Laboratory of Electronics, Cambridge, MA, March 1949 (cited on p. 47).

[92] A. Fick: „Ueber Diffusion“, *Annalen der Physik*, Vol. 170, No. 1, 59–86, April 1855 (cited on p. 29).

[93] D. J. Field: „Relations between the statistics of natural images and the response properties of cortical cells“, *Journal of the Optical Society of America A*, Vol. 4, No. 12, 2379–2394, December 1987 (cited on p. 73).

[94] G. E. Ford: „Application of inhomogeneous diffusion to image and video coding“, *Proc. 13th Asilomar Conference on Signals, Systems and Computers*, Vol. 2, 926–930, Asilomar, CA, November 1996 (cited on p. 53).

[95] E.-C. Förster, T. Löwe, S. Wenger, and M. Magnor: „RGB-guided depth map compression via compressed sensing and sparse coding“, *Proc. 31st Picture Coding Symposium*, 2, 1–4, Cairns, Australia, May 2015 (cited on pp. 62, 170).

[96] J. Fourier: *Théorie Analytique de la Chaleur*, Firmin Didot, Père et Fils, Paris, 1822 (cited on p. 29).

[97] P. Fränti and O. Nevalainen: „Compression of binary images by composite methods based on block coding“, *Journal of Visual Communication and Image Representation*, Vol. 6, No. 4, 366–377, December 1995 (cited on pp. 112, 115).

[98] I. Galić, B. Zovko-Cihlar, and S. Rimac-Drlje: „Computer image quality selection between JPEG, JPEG 2000 and PDE compression“, *Proc. 19th International Conference on Systems, Signals and Image Processing*, 437–441, Vienna, Austria, April 2012 (cited on p. 24).

[99] 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*, Vol. 31, No. 2–3, 255–269, July 2008 (cited on pp. 38, 55, 56, 69, 145, 194).

[100] 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 (Eds.): *Variational, Geometric and Level-Set Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 3752, 37–48, Springer, Berlin, 2005 (cited on pp. 2, 38, 54, 55, 102, 104, 145).

[101] Q. Gao: „Low bit rate video compression using inpainting PDEs and optic flow“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2008 (cited on pp. 59, 146).

[102] F. Garcia-Vilchez, J. Muñoz-Mari, M. Zortea, V. G. I. Blanes, G. Camps-Valls, A. Plaza, and J. Serra-Sagristà: „On the impact of lossy compression on hyperspectral image classification and unmixing“, *IEEE Geoscience and Remote Sensing Letters*, Vol. 8, No. 2, 253–257, March 2011 (cited on p. 193).

[103] J. Gautier, O. Le Meur, and C. Guillemot: „Efficient depth map compression based on lossless edge coding and diffusion“, *Proc. 29th Picture Coding Symposium*, 81–84, Kraków, Poland, May 2012 (cited on pp. 55, 58, 102, 130, 169).

[104] G. Gerig, O. Kübler, R. Kikinis, and F. A. Jolesz: „Nonlinear anisotropic filtering of MRI data“, *IEEE Transactions on Medical Imaging*, Vol. 11, No. 2, 221–232, June 1992 (cited on pp. 72, 87, 90).

[105] A. Gersho and R. M. Gray: *Vector Quantization and Signal Compression*, The Springer International Series in Engineering and Computer Science, Vol. 159, Springer, New York, 1992 (cited on p. 114).

[106] S. Golomb: „Run-length encodings“, *IEEE Transactions on Information Theory*, Vol. 12, No. 3, 399–401, July 1966 (cited on pp. 49, 156, 251).

[107] R. Gomathi and A. V. A. Kumar: „A multiresolution image completion algorithm for compressing digital color images“, *Journal of Applied Mathematics*, Vol. 2014, Article ID 757318, March 2014 (cited on p. 57).

[108] Google Corporation: *Lossy Compressed Image Formats Study*, 2011, Available at `https://developers.google.com/speed/webp/docs/webp_study`, visited on October 4, 2015 (cited on pp. 182, 192).

[109] T. Graham: „A short account of experimental researches on the diffusion of gases through each other, and their separation by mechanical means“, *Quarterly Journal of Science, Literature and Art*, Vol. 27, 74–83, July 1829 (cited on p. 28).

[110] P. Grattoni and A. Guiducci: „Contour coding for image description“, *Pattern Recognition Letters*, Vol. 11, No. 2, 95–105, February 1990 (cited on p. 57).

[111] V. V. Grebenik: „Fast hybrid fractal image compression algorithm for color images“, *Proc. 4th IEEE International Conference on Signal Processing*, Vol. 1, 804 –806, Beijing, October 1998 (cited on p. 62).

[112] J. B. Greer, A. L. Bertozzi, and G. Sapiro: „Fourth order partial differential equations on general geometries“, *Journal of Computational Physics*, Vol. 216, No. 1, 216–246, January 2006 (cited on p. 169).

[113] S. Grewenig, J. Weickert, A. Bruhn, P. Gwosdek, O. Demetz, and C. Schroers: *fed: Fast Explicit Diffusion for C/C++*, 2007, Available at `http://www.mia.uni-saarland.de/Research/SC_FED.shtml`, visited on December 13, 2015 (cited on pp. 43, 247).

[114] 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 (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 6376, 533–542, Springer, Berlin, 2010 (cited on pp. 43, 109, 147, 174).

[115] H. Grossauer and O. Scherzer: „Using the complex Ginzburg–Landau equation for digital inpainting in 2D and 3D“, In L. D. Griffin and M. Lillholm (Eds.): *Scale-Space Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 2695, 225–236, Springer, Berlin, 2003 (cited on p. 36).

[116] C. Guillemot and O. Le Meur: „Image inpainting : Overview and recent advances“, *IEEE Signal Processing Magazine*, Vol. 31, No. 1, 127–144, January 2014 (cited on p. 130).

[117] W. Hackbusch: *Multigrid Methods and Applications*, Springer Series in Computational Mathematics, Vol. 4, Springer, Berlin, 1985 (cited on p. 149).

[118] J. M. Hammersley and P. Clifford: *Markov Fields on Finite Graphs and Lattices*, unpublished, scan available at `http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf`, 1971 (cited on p. 76).

[119] D. Hankerson, G. Harris, and P. Johnson Jr.: *Introduction to Information Theory and Data Compression*, 2nd Edition, Chapman & Hall/CRC, New York, 2003 (cited on p. 47).

[120] J. Hays and A. A. Efros: „Scene completion using millions of photographs“, *ACM Transactions on Graphics*, Vol. 26, No. 3, 4.1–4.8, July 2007 (cited on p. 131).

[121] K. He and J. Sun: „Image completion approaches using the statistics of similar patches“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 12, 2423–2435, December 2014 (cited on p. 94).

[122] N. Hees, C. Williams, and G. Hinton: „Learning generative texture models with extended Fields-of-Experts“, *Proc. 20th British Machine Vision Conference*, 115.1–115.11, London, UK, August 2009 (cited on p. 130).

[123] C. Höcker and G. Fehmers: „Fast structural interpretation with structure-oriented filtering“, *The Leading Edge*, Vol. 21, No. 3, 238–243, March 2002 (cited on p. 69).

[124] 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 (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8081, 151–164, Springer, Berlin, 2013 (cited on pp. 8, 56, 64, 96, 101–103, 105, 115, 117).

[125] 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“, To appear in: *Variational Methods in Image Analysis*, De Gruyter, Berlin, 2016 (cited on pp. 35, 38, 59, 67, 102, 106, 108, 109, 188).

[126] L. Hoeltgen and J. Weickert: „Why does non-binary mask optimisation work for diffusion-based image compression?“, In X.-C. Tai, E. Bae, T. Chan, and M. Lysaker (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 85–98, Springer, Berlin, 2015 (cited on pp. 67, 105, 109).

[127] 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 (Eds.): *Scale-Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 7893, 319–330, Springer, Berlin, 2013 (cited on pp. 3, 4, 55, 58, 102, 130, 169).

[128] 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 (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 169–182, Springer, Berlin, 2015 (cited on pp. 121, 192).

[129] T. Horiuchi and S. Tominaga: „Color image coding by colorization approach“, *EURASIP Journal on Image and Video Processing*, Vol. 2008, Article ID 158273, January 2008 (cited on p. 169).

[130] B. Horn and B. Schunck: „Determining optical flow“, *Artificial Intelligence*, Vol. 17, 185–203, August 1981 (cited on p. 35).

[131] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge: „The emerging JBIG2 standard“, *IEEE Transactions on Circuits, Systems and Video Technology*, Vol. 8, No. 7, 838–848, November 1998 (cited on pp. 50, 111, 116).

[132] M. Hua, X. Bie, M. Zhang, and W. Wang: „Edge-aware gradient domain optimization framework for image filtering by local propagation“, *Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2838–2845, Columbus, OH, June 2014 (cited on p. 169).

[133] J. Huang and D. Mumford: *Image Statistics for the British Aerospace Segmented Database*, Technical Report, Divison of Applied Math, Brown University, Providence, RI, April 1999 (cited on p. 74).

[134] J. Huang and D. Mumford: „Statistics of natural images and models“, *Proc. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, 1541–1547, Fort Collins, CO, June 1999 (cited on p. 73).

[135] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu: „An adaptive edge detection based colorization algorithm and its applications“, *Proc. 13th Annual ACM International Conference on Multimedia*, 351–354, Singapore, November 2005 (cited on p. 168).

[136] D. A. Huffman: „A method for the construction of minimum redundancy codes“, *Proceedings of the IRE*, Vol. 40, No. 9, 1098–1101, September 1952 (cited on pp. 47, 109).

[137] R. Hummel and R. Moniot: „Reconstructions from zero-crossings in scale space“, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 37, No. 12, 2111–2130, December 1989 (cited on p. 57).

[138] F. Huth: „Shape Coding with Quadrupoles“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2011 (cited on pp. 59, 193).

[139] M. Hutter: *50000€ Prize for Compressing Human Knowledge*, 2006, Available at `http://prize.hutter1.net/`, visited on August 17, 2015 (cited on p. 48).

[140] T. Iijima: „Basic theory of pattern observation“, *Papers of Technical Group on Automata and Automatic Control*, In Japanese, IECE, Kyoto, Japan, 1959 (cited on pp. 3, 32).

[141] T. Iijima: „Basic theory on normalization of pattern (in case of typical one-dimensional pattern)“, *Bulletin of the Electrotechnical Laboratory*, Vol. 26, In Japanese, 368–388, January 1962 (cited on pp. 3, 32, 71, 87, 90, 104).

[142] R. Irony, D. Cohen-Or, and D. Lischinski: „Colorization by example“, *Proc. 16th Eurographics Symposium on Rendering*, 201–210, Konstanz, Germany, June 2005 (cited on p. 168).

[143] ISO/IEC: *Information Technology – Coding of Audio-visual Objects – Part 10: Advanced Video Coding*, Standard, ISO/IEC 14496-10, 2003 (cited on pp. v, vii, 53).

[144] F. Jäger: „Contour-based segmentation and coding for depth map compression“, *Proc. 2011 IEEE Conference on Visual Communications and Image Processing*, 1–4, Tainan, Taiwan, November 2011 (cited on p. 62).

[145] P. Johansen, S. Skelboe, K. Grue, and J. D. Andersen: „Representing signals by their toppoints in scale space“, *Proc. Eighth International Conference on Pattern Recognition*, 215–217, Paris, France, October 1986 (cited on p. 59).

[146] Joint Bi-level Image Experts Group: *Information Technology – Progressive Lossy/Lossless Coding of Bi-level Images*, Standard, International Organization for Standardization, 1993 (cited on pp. 50, 111, 115).

[147] S. H. Kang and R. March: „Variational models for image colorization via chromaticity and brightness decomposition“, *IEEE Transactions on Image Processing*, Vol. 16, No. 9, 2251–2261, September 2007 (cited on p. 168).

[148] G. Kanizsa: „Margini quasi-percettivi in campi con stimolazione omogenea“, *Rivista di Psicologia*, Vol. 49, No. 1, 7–30, March 1955 (cited on p. 35).

[149] F. M. W. Kanters, M. Lillholm, R. Duits, B. J. P. Jansen, B. Platel, L. Florack, and B. M. ter Haar Romeny: „On image reconstruction from multiscale top points“, In R. Kimmel, N. Sochen, and J. Weickert (Eds.): *Scale Space and PDE Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 3459, 431–439, Springer, Berlin, 2005 (cited on p. 59).

[150] L. Kaufhold: „Variational Methods for the Colourisation of Greyscale Images“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on p. 183).

[151] N. Kaur: „A review of region-of-interest coding techniques of JPEG2000“, *IJCA Special Issue on Computational Science - New Dimensions & Perspectives*, Vol. 1, 35–40, January 2011 (cited on p. 164).

[152] B.-J. Kim and W. Pearlman: „An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)“, *Proc. 1997 Data Compression Conference*, 251–260, Snowbird, UT, March 1997 (cited on pp. 61, 62).

[153] A. Kolaman and O. Yadid-Pecht: „Quaternion structural similarity: A new quality index for color images“, *IEEE Transactions on Image Processing*, Vol. 21, No. 4, 1526–1536, December 2011 (cited on p. 24).

[154] V. Konushin and V. Vezhnevets: „Interactive image colorization and recoloring based on coupled map lattices“, *Proc. 16th International Conference on Computer Graphics and Applications*, 231–234, Akademgorodok, Russia, July 2006 (cited on pp. 168, 169).

[155] I. Kopilovic and T. Szirányi: „Artifact reduction with diffusion preprocessing for image compression“, *Optical Engineering*, Vol. 44, No. 2, Article ID 27003, February 2005 (cited on p. 53).

[156] H. Köstler, M. Stürmer, C. Freundl, and U. Rüde: *PDE Based Video Compression in Real Time*, Technical Report No. 07-11, Lehrstuhl für Informatik 10, University Erlangen–Nürnberg, Germany, 2007 (cited on pp. 6, 59, 102, 146, 189).

[157] K. Krajsek: „Probabilistic Models and Estimation: Theoretical Concepts for Variational Approaches to Computer Vision“, Doctoral Thesis, Universität Frankfurt, Frankfurt, Germany, 2015 (cited on p. 75).

[158] K. Krajsek and H. Scharr: „Diffusion filtering without parameter tuning: Models and inference tools“, *Proc. 2010 IEEE Conference on Computer Vision and Pattern Recognition*, 2536–2543, San Francisco, CA, June 2010 (cited on pp. 74, 81, 97).

[159] K. Kunisch and T. Pock: „A bilevel optimization approach for parameter learning in variational models“, *SIAM Journal on Imaging Sciences*, Vol. 6, No. 2, 938–983, May 2013 (cited on p. 75).

[160] K. Kunow: „Current status of digitisation in german (TV) households“, In L. Keimburg and K. Kunow (Eds.): *Digitisation: All in Flux – New Forms and Old Patterns*, Vol. 8081, 20–29, die medienanstalten – ALM GbR, Berlin, 2013 (cited on p. 2).

[161] M. Kunt and O. Johnsen: „Block coding of graphics: A tutorial review“, *Proceedings of the IEEE*, Vol. 68, No. 7, 770–786, July 1980 (cited on p. 112).

[162] M. Kunt, A. Ikonomopoulos, and M. Kocher: „Second-generation image-coding techniques“, *Proceedings of the IEEE*, Vol. 73, No. 4, 549–574, April 1985 (cited on p. 61).

[163] B. Kurt, M. Gökmen, and A. K. Jain: „Image compression based on centipede model“, In A. Del Bimbo (Ed.): *Proc. Ninth International Conference on Image Analysis and Processing*, Lecture Notes in Computer Science, Vol. 1310, 303–310, Springer, Berlin, 1997 (cited on p. 57).

[164] A. Levin, D. Lischinski, and Y. Weiss: „Colorization using optimization“, *Proc. SIGGRAPH 2004*, 689–694, New York, NY, August 2004 (cited on pp. 167–169, 173, 189, 247).

[165] A. Levin, A. Zomet, and Y. Weiss: „Learning how to inpaint from global image statistics“, *Proc. Ninth International Conference on Computer Vision*, Vol. 1, 305–312, Nice, France, October 2003 (cited on p. 94).

[166] Y. Li, M. Sjostrom, U. Jennehag, and R. Olsson: „A scalable coding approach for high quality depth image compression.“, *Proc. 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video*, 1–4, Zurich, Switzerland, October 2012 (cited on pp. 3, 55, 58, 102, 130, 169).

[167] M. Lillholm, M. Nielsen, and L. D. Griffin: „Feature-based image analysis“, *International Journal of Computer Vision*, Vol. 52, No. 2, 73–95, May 2003 (cited on p. 59).

[168] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski: „Interactive local adjustment of tonal values“, *ACM Transactions on Graphics*, Vol. 25, No. 3, 646–653, July 2006 (cited on p. 168).

[169] D. Liu, X. Sun, and F. Wu: „Edge-based inpainting and texture synthesis for image compression“, *Proc. 2007 IEEE International Conference on Multimedia and Expo*, 1443–1446, Beijing, China, July 2007 (cited on p. 59).

[170] 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*, Vol. 17, No. 10, 1273–1286, October 2007 (cited on p. 131).

[171] L. Liu and G. Fan: „A new JPEG2000 region-of-interest image coding method: Partial significant bitplanes shift“, *IEEE Signal Processing Letters*, Vol. 10, No. 2, 35–38, February 2003 (cited on p. 164).

[172] R. Liu, J. Cao, Z. Lin, and S. Shan: „Adaptive partial differential equation learning for visual saliency detection“, *Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition*, 3866–3873, Columbus, OH, June 2014 (cited on p. 75).

[173] S. P. Lloyd: „Least squares quantization in PCM“, *IEEE Transactions on Information Theory*, Vol. 28, No. 2, 129–137, March 1982 (cited on p. 113).

[174] B. F. Logan, Jr.: „Information in the zero crossings of bandpass signals“, *Bell System Technical Journal*, Vol. 56, 487–510, April 1977 (cited on p. 57).

[175] A. Loza, L. Mihaylova, N. Canagarajah, and D. Bull: „Structural similarity-based object tracking in video sequences“, *Proc. 9th International Conference on Information Fusion*, 1–6, Florence, Italy, July 2006 (cited on p. 20).

[176] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum: „Natural image colorization“, *Proc. 18th Eurographics Conference on Rendering Techniques*, 309–320, Grenoble, France, 2007 (cited on pp. 168, 169).

[177] R. Lund: „3-D Data Compression with Homogeneous Diffusion“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2011 (cited on p. 60).

[178] M. Mahoney: *Adaptive weighing of context models for lossless data compression*, Technical Report No. CS-2005-16, Florida Institute of Technology, Melbourne, FL, December 2005 (cited on pp. 48, 109, 116, 138, 139, 179, 181, 191, 248).

[179] M. Mahoney: *Data Compression Explained*, Dell Inc., Round Rock, 2010 (cited on p. 48).

[180] M. Mahoney: *Data Compression Programs*, 2009, Available at `http://mattmahoney.net/dc/`, visited on December 13, 2015 (cited on pp. 48, 49).

[181] M. Mainberger and J. Weickert: „Edge-based image compression with homogeneous diffusion", In X. Jiang and N. Petkov (Eds.): *Computer Analysis of Images and Patterns*, Lecture Notes in Computer Science, Vol. 5702, 476–483, Springer, Berlin, 2009 (cited on pp. 55, 56, 58, 101).

[182] 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 (Eds.): *Advances in Visual Computing, Part II*, Lecture Notes in Computer Science, Vol. 7432, 219–228, Springer, Berlin, 2012 (cited on pp. 6, 56, 61, 190).

[183] M. Mainberger, A. Bruhn, J. Weickert, and S. Forchhammer: „Edge-based compression of cartoon-like images with homogeneous diffusion", *Pattern Recognition*, Vol. 44, No. 9, 1859–1873, September 2011 (cited on pp. 49, 55, 56, 58, 102, 111, 112, 130, 169).

[184] 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. Bruckstein, B. ter Haar Romeny, A. Bronstein, and M. Bronstein (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 6667, 26–37, Springer, Berlin, 2011 (cited on pp. 3, 4, 56, 58, 64, 67, 96, 101–103, 105, 106, 108, 114–119, 147, 193).

[185] F. Malgouyres and F. Guichard: „Edge direction preserving image zooming: A mathematical and numerical analysis", *SIAM Journal on Numerical Analysis*, Vol. 39, No. 1, 1–37, January 2001 (cited on p. 35).

[186] S. Mallat and S. Zhong: „Characterisation of signals from multiscale edges", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 7, 720–732, July 1992 (cited on p. 57).

[187] R. Manniesing, M. A. Viergever, and W. J. Niessen: „Vessel enhancing diffusion: A scale space representation of vessel structures", *Medical Image Analysis*, Vol. 10, 815–825, December 2006 (cited on p. 69).

[188] D. Martin, C. Fowlkes, D. Tal, and J. Malik: „A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics", *Proc. Eigth International Conference on Computer Vision*, 416–423, Vancouver, Canada, July 2001 (cited on pp. 73, 74, 81, 90, 140, 173).

[189] G. N. N. Martin: „Range encoding: An algorithm for removing redundancy from a digitized message", *Proc. Video & Data Recording Conference*, Southampton, UK, July 1979 (cited on p. 48).

[190] S. Masnou and J.-M. Morel: „Level lines based disocclusion“, *Proc. 1998 IEEE International Conference on Image Processing*, Vol. 3, 259–263, Chicago, IL, October 1998 (cited on p. 35).

[191] J. Max: „Quantizing for minimum distortion“, *IRE Transactions on Information Theory*, Vol. 6, No. 1, 7–12, March 1960 (cited on p. 113).

[192] H. Mehrer and N. A. Stolwijk: „Heroes and highlights in the history of diffusion“, *Diffusion Fundamentals*, Vol. 11, No. 1, 1–32, August 2009 (cited on p. 28).

[193] E. Meijering: „A chronology of interpolation: From ancient astronomy to modern signal and image processing“, *Proceedings of the IEEE*, Vol. 90, No. 3, 319–342, March 2002 (cited on p. 35).

[194] A. Meister: *Numerik linearer Gleichungssysteme*, Vieweg, Braunschweig, 1999 (cited on p. 44).

[195] A. Moffat: „Implementing the PPM data compression scheme“, *IEEE Transactions on Communications*, Vol. 38, No. 11, 1917–1921, November 1990 (cited on p. 48).

[196] S. A. Mohamed and M Fahmy: „Binary image compression using efficient partitioning into rectangular regions“, *IEEE Transactions on Communications*, Vol. 43, No. 5, 1888–1893, May 1995 (cited on pp. 112, 115).

[197] Y. Morvan, P. 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 (Eds.): *Stereoscopic Displays and Virtual Reality Systems XIII*, Proceedings of SPIE, Vol. 6055, Article ID 60550K, SPIE Press, Bellingham, WA, 2006 (cited on p. 62).

[198] K. T. Mullen: „The contrast sensitivity of human colour vision to red-green and blue-yellow chromatic gratings.“, *The Journal of Physiology*, Vol. 359, 381–400, February 1985 (cited on pp. 26, 167).

[199] D. Mumford and J. Shah: „Boundary detection by minimizing functionals“, *Proc. First IEEE Conference on Computer Vision and Pattern Recognition*, 22–26, IEEE Computer Society Press, San Francisco, CA, June 1985 (cited on p. 75).

[200] J. Murray and W. van Ryper: *Encyclopedia of Graphics File Formats*, O’Reilly, Sebastopol, 1996 (cited on p. 49).

[201] 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*, Vol. 8, 565–593, September 1986 (cited on p. 69).

[202] National Electrical Manufacturers Association: *Digital Imaging and Communications in Medicine (DICOM) – Part 5 Data Structures and Encoding*, PS 3.5-2004, 2004 (cited on p. 50).

[203] P. Ndjiki-Nya, D. Doshkov, H. Kaprykowsky, F. Zhang, D. Bull, and T. Wiegand: „Perception-oriented video coding based on image analysis and completion: A review“, *Signal Processing: Image Communication*, Vol. 27, No. 6, 579–594, February 2012 (cited on p. 193).

[204] F. Nedwed: „A Probabilistic Approach to Image Compression Using Subdivisions“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2015 (cited on p. 128).

[205] 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 (cited on p. 61).

[206] H. Noda and M. Niimi: „Local MAP estimation for quality improvement of compressed color images“, *Pattern Recognition*, Vol. 44, No. 4, 788–793, October 2011 (cited on p. 170).

[207] H. Noda, J. Korekuni, and M. Niimi: „A colorization algorithm based on local map estimation“, *Pattern Recognition*, Vol. 39, No. 11, 2212–2217, July 2006 (cited on p. 170).

[208] H. Noda, N. Takao, and M. Niimi: „Colorization in YCbCr space and its application to improve quality of JPEG color images“, *Proc. 14th IEEE International Conference on Image Processing*, Vol. 4, 385–388, San Antonio, TX, September 2007 (cited on p. 170).

[209] P. Ochs, Y. Chen, T. Brox, and T. Pock: „iPiano: Inertial proximal algorithm for nonconvex optimization“, *SIAM Journal on Applied Mathematics*, Vol. 7, No. 2, 1388–1419, June 2014 (cited on pp. 56, 64).

[210] K. Okarma: „Colour image quality assessment using structural similarity index and singular value decomposition“, In L. Bolc, J. Kulikowski, and K. Wojciechowski (Eds.): *Computer Vision and Graphics*, Lecture Notes in Computer Science, Vol. 5337, 55–65, Springer, Berlin, 2006 (cited on p. 24).

[211] A. Olmos and F. Kingdom: „A biologically inspired algorithm for the recovery of shading and reflectance images.“, *Perception*, Vol. 33, No. 12, 1463–1473, December 2004 (cited on pp. 74, 81).

[212] B. Ou, X. Li, Y. Zhao, and R. Ni: „Reversible data hiding based on PDE predictor“, *Journal of Systems and Software*, Vol. 86, No. 10, 2700–2709, June 2013 (cited on p. 61).

[213] J. S. Pawadshetty and J. W. Bakal: „JPEG 2000 region of interest coding methods“, *International Journal of Engineering Research and Applications*, Vol. 3, No. 1, 1184–1188, January 2013 (cited on p. 164).

[214] K. S. Pedersen, R. Duits, and M. Nielsen: „On α kernels, Lévy processes, and natural image statistics“, In R. Kimmel, N. Sochen, and J. Weickert (Eds.): *Scale Space and PDE Methods in Computer Vision*, Lecture Notes in Computer Science, 468–479, Springer, Berlin, 2005 (cited on p. 74).

[215] W. B. Pennebaker and J. L. Mitchell: *JPEG: Still Image Data Compression Standard*, Springer, New York, 1992 (cited on pp. v, vii, 2–5, 50, 53, 164, 167, 169, 187).

[216] P. Pérez: „Markov random fields and images“, *CWI Quarterly*, Vol. 11, No. 4, 413–437, September 1998 (cited on p. 77).

[217] P. Perona and J. Malik: „Scale space and edge detection using anisotropic diffusion“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, 629–639, July 1990 (cited on pp. 32, 34, 72, 75, 87, 89).

[218] P. Peter: „Three-dimensional data compression with anisotropic diffusion“, In J. Weickert, M. Hein, and B. Schiele (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8142, 231–236, Springer, Berlin, 2013 (cited on pp. 60, 130, 169, 193).

[219] P. Peter and J. Weickert: „Colour image compression with anisotropic diffusion“, *Proc. 21st IEEE International Conference on Image Processing*, 4822–4862, Paris, France, October 2014 (cited on pp. 102, 183).

[220] P. Peter and J. Weickert: „Compressing images with diffusion- and exemplar-based inpainting“, In J.-F. Aujol, M. Nikolova, and N. Papadakis (Eds.): *Scale-Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 9087, 154–165, Springer, Berlin, 2015 (cited on pp. 144, 169, 185).

[221] P. Peter, C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert: „Beyond pure quality: Progressive modes, region of interest coding, and real time video decoding for PDE-based image compression.“, *Journal of Visual Communication and Image Representation*, Vol. 31, No. 4, 253–265, August 2015 (cited on pp. 161, 165).

[222] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert: *Evaluating the True Potential of Diffusion-Based Inpainting in a Compression Context*, Technical Report No. 373, Department of Mathematics, Saarland University, Saarbrücken, Germany, January 2016 (cited on p. 128).

[223] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert: „From optimised inpainting with linear PDEs towards competitive image compression codecs“, To appear in T. Bräunl, B. McCane, M. Rivers, and X. Yu (Eds.): *Advances in Image and Video Technology*, Lecture Notes in Computer Science, Springer, Berlin, 2016 (cited on p. 128).

[224] P. Peter, J. Weickert, A. Munk, T. Krivobokova, and H. Li: „Justifying tensor-driven diffusion from structure-adaptive statistics of natural images“, In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 263–277, Springer, Berlin, 2015 (cited on p. 100).

[225] P. Peter, L. Kaufhold, and J. Weickert: *Turning Diffusion-based Image Colourisation into Efficient Colour Compression*, Technical Report No. 370, Department of Mathematics, Saarland University, Saarbrücken, Germany, December 2015 (cited on p. 183).

[226] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli: „Adaptive Wiener denoising using a Gaussian scale mixture model in the wavelet domain“, *Proc. 2001 IEEE International Conference on Image Processing*, Vol. 2, 37–40, Thessaloniki, Greece, October 2001 (cited on p. 74).

[227] T. Pouli, E. Reinhard, and D. W. Cunningham: *Image Statistics in Visual Computing*, CRC Press, Boca Raton, 2013 (cited on p. 74).

[228] W. K. Pratt: *Digital Image Processing: PIKS Inside*, 3rd Edition, Wiley, New York, 2001 (cited on p. 171).

[229] M. Puhl: „Edge Based Image Compression with Mixed Boundary Conditions“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2012 (cited on p. 58).

[230] Y. Qu, T.-T. Wong, and P.-A. Heng: „Manga colorization“, *ACM Transactions on Graphics*, Vol. 25, No. 3, 1214–1220, July 2006 (cited on p. 169).

[231] M. H. Quang, S. H. Kang, and T. M. Le: „Image and video colorization using vector-valued reproducing kernel Hilbert spaces“, *Journal of Mathematical Imaging and Vision*, Vol. 37, No. 1, 49–65, May 2010 (cited on pp. 168, 169).

[232] S. D. Rane, G. Sapiro, and M. Bertalmío: „Structure and texture fillingin of missing image blocks in wireless transmission and compression applications“, *IEEE Transactions on Image Processing*, Vol. 12, No. 3, 296–302, March 2003 (cited on p. 131).

[233] M. Reid, R. J. Millar, and N. D. Black: „Second-generation image coding: An overview“, *ACM Computing Surveys*, Vol. 29, No. 1, 3–29, March 1997 (cited on p. 61).

[234] J. J. Rissanen: „Generalized Kraft inequality and arithmetic coding“, *IBM Journal of Research and Development*, Vol. 20, No. 3, 198–203, May 1976 (cited on pp. 48, 109).

[235] J. J. Rissanen and G. G. Langdon Jr.: „Arithmetic coding“, *IBM Journal of Research and Development*, Vol. 23, No. 2, 149–162, March 1979 (cited on p. 48).

[236] D. Rotem and Y. Zeevi: „Image reconstruction from zero-crossings“, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 34, 1269–1277, October 1986 (cited on p. 57).

[237] S. Roth and M. J. Black: „Fields of experts“, *International Journal of Computer Vision*, Vol. 82, No. 2, 205–229, April 2009 (cited on pp. 74, 93).

[238] S. Roth and M. J. Black: „Fields of experts: A framework for learning image priors“, *Proc. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2, 860–867, San Diego, CA, June 2005 (cited on pp. 74–77, 130, 190).

[239] A. Roussos and P. Maragos: „Tensor-based image diffusions derived from generalizations of the total variation and Beltrami functionals“, *Proc. 17th IEEE International Conference on Image Processing*, 4141–4144, Hong Kong, September 2010 (cited on pp. 72, 87–90, 94, 96, 237, 239, 240).

[240] L. I. Rudin, S. Osher, and E. Fatemi: „Nonlinear total variation based noise removal algorithms“, *Physica D*, Vol. 60, No. 1, 259–268, November 1992 (cited on p. 38).

[241] C. Saloma and P. Haeberli: „Two-dimensional image reconstruction from Fourier coefficients computed directly from zero crossings“, *Applied Optics*, Vol. 32, No. 17, 3092–3093, April 1993 (cited on p. 57).

[242] M. Sampat, Z. Wang, S. Gupta, A. Bovik, and M. Markey: „Complex wavelet structural similarity: A new image similarity index“, *IEEE Transactions on Image Processing*, Vol. 18, No. 11, 2385–2401, June 2009 (cited on p. 24).

[243] M. Sampat, Z. Wang, M. Markey, G. Whitman, T. Stephens, and A. Bovik: „Measuring intra- and inter-observer agreement in identifying and localizing structures in medical images“, *Proc. 13th IEEE International Conference on Image Processing*, 81–84, Atlanta, GA, October 2006 (cited on p. 20).

[244] M. Sarkis, W. Zia, and K. Diepold: „Fast depth map compression and meshing with compressed tritree“, In H. Zha, R. Taniguchi, and S. Maybank (Eds.): *Computer Vision – ACCV 2009*, Lecture Notes in Computer Science, 44–55, Springer, Berlin, 2010 (cited on p. 62).

[245] K. Sayood: *Introduction to Data Compression*, 3rd Edition, Morgan Kaufmann, San Francisco, 2005 (cited on p. 49).

[246] H. Scharr, M. J. Black, and H. W. Haussecker: „Image statistics and anisotropic diffusion“, *Proc. Ninth International Conferance on Computer Vision*, Vol. 2, 840–847, Nice, France, 2003 (cited on pp. 72, 74, 86, 87, 89, 90, 93).

[247] A. Scheer: „Entropy Coding for PDE-Based Image Compression“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2014 (cited on pp. 61, 110, 191).

[248] O. Scherzer and J. Weickert: „Relations between regularization and diffusion filtering“, *Journal of Mathematical Imaging and Vision*, Vol. 12, No. 1, 43–63, February 2000 (cited on pp. 72, 79, 87, 88, 90).

[249] 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 (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 7476, 438–447, Springer, Berlin, 2012 (cited on pp. 59, 62, 145).

[250] C. Schmaltz, J. Weickert, and A. Bruhn: „Beating the quality of JPEG 2000 with anisotropic diffusion“, In J. Denzler, G. Notni, and H. Süße (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 5748, 452–461, Springer, Berlin, 2009 (cited on pp. v, vii, 2, 55, 56, 64–66, 68, 129, 130, 138, 140, 145, 147, 167, 169, 178).

[251] C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert: „Progressive modes in PDE-based image compression.“, *Proc. 30th Picture Coding Symposium*, 233–236, San Jose, CA, December 2013 (cited on pp. 6, 61, 102, 144, 190).

[252] 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*, Vol. 108, No. 3, 222–240, July 2014 (cited on pp. 27, 38, 45, 49, 55, 56, 59, 64, 65, 69, 97, 101, 102, 104, 106, 109, 110, 118, 130, 134, 135, 138, 139, 145, 147, 155, 167, 189, 192, 194).

[253] M. Schneider: „Image Compression with Homogeneous Diffusion Inpainting on Level Lines“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on p. 61).

[254] M. Schneider: „Image Compression with Sparse Gradient Data“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2015 (cited on pp. 61, 193).

[255] M. Schwinn: „Patch-Based Inpainting and Compression“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on p. 59).

[256] J. Serra-Sagristà and F. Aulí-Llinàs: „Remote sensing data compression“, In M. Graña and R. J. Duro (Eds.): *Computational Intelligence for Remote Sensing*, Studies in Computational Intelligence, Vol. 133, 27–61, Springer, Berlin, 2008 (cited on p. 192).

[257] J. Seward: *bzip2 and libbzip2, Version 1.0.5, A Program and Library for Data Compression*, 2008, Available at `http://www.bzip.org/.`, visited on December 15, 2015 (cited on p. 49).

[258] C. E. Shannon: „A mathematical theory of communication – Part 1“, *Bell System Technical Journal*, Vol. 27, No. 3, 379–423, July 1948 (cited on pp. 46, 47).

[259] C. E. Shannon: „A mathematical theory of communication – Part 2“, *Bell System Technical Journal*, Vol. 27, No. 4, 623–656, October 1948 (cited on p. 46).

[260] C. E. Shannon and W. Weaver: *The Mathematical Theory of Communication*, University of Illinois Press, Urbana, 1949 (cited on pp. 46, 130).

[261] H. Sheikh and A. Bovik: „Image information and visual quality“, *IEEE Transactions on Image Processing*, Vol. 15, No. 2, 430–444, February 2006 (cited on p. 24).

[262] A. Solé, V. Caselles, G. Sapiro, and F. Arandiga: „Morse description and geometric encoding of digital elevation maps“, *IEEE Transactions on Image Processing*, Vol. 13, No. 9, 1245–1262, September 2004 (cited on p. 61).

[263] A. Srivastava, A. B. Lee, E. P. Simoncelli, and S.-C. Zhu: „On advances in statistical modeling of natural images“, *Journal of Mathematical Imaging and Vision*, Vol. 18, No. 1, 17–33, January 2003 (cited on p. 74).

[264] Statistisches Bundesamt: *Ausstattung privater Haushalte mit ausgewählten Gebrauchsgütern*, Fachserie 15, Reihe 2, 2015 (cited on p. 1).

[265] J. Steil: „Shape Coding with Nonapoles“, Bachelor Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on pp. 59, 193).

[266] P. Strobach: „Quadtree-structured interframe coding of HDTV sequences“, In T. R. Hsing (Ed.): *Visual Communications and Image Processing ’88*, Proceedings of SPIE, Vol. 1001, 812–825, SPIE Press, Bellingham, WA, 1988 (cited on p. 61).

[267] P. Strobach: „Quadtree-structured recursive plane decomposition coding of images“, *IEEE Transactions on Signal Processing*, Vol. 39, No. 6, 1380–1397, June 1991 (cited on p. 61).

[268] T. Strutz: *Bilddatenkompression*, 4th Edition, Vieweg + Teubner, Braunschweig, 2009 (cited on pp. 1, 46, 113).

[269] G. J. Sullivan and R. Baker: „Efficient quadtree coding of images and video“, *IEEE Transactions on Image Processing*, Vol. 3, No. 3, 327–331, May 1994 (cited on p. 61).

[270] G. J. Sullivan and T. Wiegand: „Video compression – From concepts to the H. 264/AVC standard“, *Proceedings of the IEEE*, Vol. 93, No. 1, 18–31, January 2005 (cited on p. 53).

[271] 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, Systems and Video Technology*, Vol. 22, No. 12, 1649–1668, September 2012 (cited on p. 53).

[272] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum: „Image completion with structure propagation“, *ACM Transactions on Graphics*, Vol. 24, No. 3, 861–868, July 2005 (cited on p. 130).

[273] S. E. Süsstrunk and S. Winkler: „Color image quality on the internet“, In S. Santini and R. Schettini (Eds.): *Internet Imaging V*, Proceedings of SPIE, Vol. 5304, 118–131, SPIE Press, San Jose, CA, 2004 (cited on p. 20).

[274] D. Sỳkora, J. Buriánek, and J. Žára: „Unsupervised colorization of black-and-white cartoons“, *Proc. 3rd International Symposium on Non-Photorealistic Animation and Rendering*, 121–127, Annecy, France, June 2004 (cited on pp. 167, 168).

[275] T. Szirányi, I. Kopilovic, and B. P. Tóth: „Anisotropic diffusion as a preprocessing step for efficient image compression“, *Proc. 14th International Conference on Pattern Recognition*, Vol. 2, 1565–1567, Brisbane, Australia, August 1998 (cited on pp. 53, 54).

[276] Y.-W. Tai, J. Jia, and C.-K. Tang: „Local color transfer via probabilistic segmentation by expectation-maximization“, *Proc. 2005 IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, 747–754, San Diego, CA, June 2005 (cited on p. 168).

[277] D. S. Taubman and M. W. Marcellin (Ed.): *JPEG 2000: Image Compression Fundamentals, Standards and Practice*, Kluwer, Boston, 2002 (cited on pp. v, vii, 2, 167, 169).

[278] A. Telea: „An image inpainting technique based on the fast marching method“, *Journal of Graphics Tools*, Vol. 9, No. 1, 23–34, April 2004 (cited on p. 38).

[279] S. Toelg and T. Poggio: *Towards an Example-Based Image Compression Architecture for Video-Conferencing*, Technical Report No. AIM-1494, Massachusetts Institute of Technology, Cambridge, MA, 1994 (cited on p. 62).

[280] K. Tönnies: *Grundlagen der Bildverarbeitung*, Pearson Studium, München, 2005 (cited on p. 113).

[281] 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*, Vol. 27, No. 4, 506–516, April 2005 (cited on pp. 36, 69, 72, 87, 89, 90).

[282] D. Tschumperlé: „PDE's Based Regularization of Multivalued Images and Applications“, PhD Thesis, Université de Nice – Sophia Antipolis, Nice, France, 2002 (cited on pp. 75, 97).

[283] H. Tsuji, T. Sakatani, Y. Yashima, and N. Kobayashi: „A nonlinear spatio-temporal diffusion and its application to prefiltering in MPEG-4 video coding“, *Proc. 2002 IEEE International Conference on Image Processing*, Vol. 1, 85–88, Rochester, NY, September 2002 (cited on p. 54).

[284] H. Tsuji, S. Tokumasu, H. Takahashi, and M. Nakajima: „Spatial prefiltering scheme based on anisotropic diffusion in low-bitrate video coding“, *Systems and Computers in Japan*, Vol. 38, No. 10, 34–45, June 2007 (cited on p. 54).

[285] A. Vail: *Description of the American Electro Magnetic Telegraph: Now in Operation between the Cities of Washington and Baltimore*, J. & G.S. Gideon, Washington, 1945 (cited on p. 47).

[286] D. Vaisey and A. Gersho: „Variable block-size image coding“, *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 12, 1051–1054, Dallas, TX, April 1987 (cited on p. 61).

[287] R. A. Varga: *Matrix Iterative Analysis*, Prentice Hall, Englewood Cliffs, 1962 (cited on pp. 45, 147, 148).

[288] Z. Wang and A. C. Bovik: „Bitplane-by-bitplane shift (BbBShift) – A suggestion for JPEG2000 region of interest image coding“, *IEEE Signal Processing Letters*, Vol. 9, No. 5, 160–162, May 2002 (cited on p. 164).

[289] 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*, Vol. 13, No. 4, 600–612, April 2004 (cited on pp. 20, 21, 24, 26, 182).

[290] Z. Wang, E. Simoncelli, and A. Bovik: „Multiscale structural similarity for image quality assessment“, *Proc. 37th Asilomar Conference on Signals, Systems and Computers*, Vol. 2, 1398–1402, Asilomar, CA, November 2004 (cited on p. 24).

[291] Z. Wang, Q. Li, and X. Shang: „Perceptual image coding based on a maximum of minimal structural similarity criterion“, *Proc. 14th IEEE International Conference on Image Processing*, Vol. 2, 121–124, San Antonio, TX, October 2007 (cited on p. 20).

[292] J. Weickert: „Anisotropic diffusion filters for image processing based quality control“, In A. Fasano and M. Primicerio (Eds.): *Proc. Seventh European Conference on Mathematics in Industry*, 355–362, Teubner, Stuttgart, 1994 (cited on p. 69).

[293] J. Weickert: *Anisotropic Diffusion in Image Processing*, Teubner, Stuttgart, 1998 (cited on pp. 29–31, 40, 43, 44, 89).

[294] J. Weickert: „Nonlinear diffusion filtering“, In B. Jähne, H. Haußecker, and P. Geißler (Eds.): *Handbook on Computer Vision and Applications, Vol. 2: Signal Processing and Pattern Recognition*, 423–450, Academic Press, San Diego, CA, 1999 (cited on pp. 40, 173).

[295] J. Weickert: „Theoretical foundations of anisotropic diffusion in image processing“, In W. Kropatsch, R. Klette, F. Solina, and R. Albrecht (Eds.): *Theoretical Foundations of Computer Vision*, Computing Supplement, Vol. 11, 221–236, Springer, Vienna, Austria, 1996 (cited on pp. v, vii, 3, 34, 54, 101, 102, 129, 134).

[296] J. Weickert and T. Brox: „Diffusion and regularization of vector- and matrix-valued images“, In M. Z. Nashed and O. Scherzer (Eds.): *Inverse Problems, Image Analysis, and Medical Imaging*, Contemporary Mathematics, Vol. 313, 251–268, AMS, Providence, RI, 2002 (cited on pp. 72, 87, 89, 90).

[297] J. Weickert and M. Welk: „Tensor field interpolation with PDEs“, In J. Weickert and H. Hagen (Eds.): *Visualization and Processing of Tensor Fields*, 315–325, Springer, Berlin, 2006 (cited on p. 36).

[298] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn: „Cyclic schemes for PDE-based image analysis“, To appear in *International Journal of Computer Vision* 2016 (cited on pp. 43, 45, 148, 189).

[299] J. Weickert, M. Welk, and M. Wickert: „L2-stable nonstandard finite differences for anisotropic diffusion“, In A. Kuijper, K. Bredies, T. Pock, and H. Bischof (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 7893, 380–391, Springer, Berlin, 2013 (cited on pp. 40–43, 81, 147, 174, 247).

[300] J. Weickert, S. Ishikawa, and A. Imiya: „Linear scale-space has first been proposed in Japan“, *Journal of Mathematical Imaging and Vision*, Vol. 10, No. 3, 237–252, May 1999 (cited on p. 32).

[301] M. Weinberger, G. Seroussi, and G. Sapiro: „The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS“, *IEEE Transactions on Image Processing*, Vol. 9, No. 8, 1309–1324, August 2000 (cited on p. 50).

[302] P. Weinzaepfel, H. Jégou, and P. Pérez: „Reconstructing an image from its local descriptors“, *Proc. 2011 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 337–344, Colorado Springs, CO, November 2011 (cited on pp. 36, 59).

[303] T. A. Welch: „A technique for high-performance data compression“, *Computer*, Vol. 17, No. 6, 8–19, June 1984 (cited on p. 49).

[304] M. Welk, J. Weickert, and G. Steidl: „From tensor-driven diffusion to anisotropic wavelet shrinkage“, In H. Bischof, A. Leonardis, and A. Pinz (Eds.): *Computer Vision – ECCV 2006, Part I*, Lecture Notes in Computer Science, Vol. 3951, 391–403, Springer, Berlin, 2006 (cited on p. 40).

[305] M. Welk, G. Steidl, and J. Weickert: „Locally analytic schemes: A link between diffusion filtering and wavelet shrinkage“, *Applied and Computational Harmonic Analysis*, Vol. 24, 195–224, March 2008 (cited on pp. 40, 41, 80, 81).

[306] T. Welsh, M. Ashikhmin, and K. Mueller: „Transferring color to greyscale images“, *ACM Transactions on Graphics*, Vol. 21, No. 3, 277–280, July 2002 (cited on p. 168).

[307] H. Werner: „Studies on contour“, *The American Journal of Psychology*, Vol. 47, No. 1, 40–64, January 1935 (cited on pp. 35, 57).

[308] A. P. Witkin: „Scale-space filtering“, *Proc. Eighth International Joint Conference on Artificial Intelligence*, Vol. 2, 945–951, Karlsruhe, Germany, August 1983 (cited on p. 32).

[309] I. H. Witten, R. M. Neal, and J. G. Cleary: „Arithmetic coding for data compression“, *Communications of the ACM,* Vol. 30, No. 6, 520–540, June 1987 (cited on p. 48).

[310] Y. Wu, H. Zhang, Y. Sun, and H. Guo: „Two image compression schemes based on image inpainting“, *Proc. 2009 International Joint Conference on Computational Sciences and Optimization*, 816–820, Sanya, China, April 2009 (cited on pp. 55, 57).

[311] 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 (Ed.): *Advanced Signal Processing Algorithms, Architectures, and Implementations XVII,* Proceedings of SPIE, Vol. 6697, SPIE Press, Bellingham, WA, 2007 (cited on p. 61).

[312] W. Xiong, X. Y. Sun, F. Wu, and S. P. Li: „Image coding with parameter-assistant inpainting“, *Proc. 14th IEEE International Conference on Image Processing*, Vol. 2, 369–372, San Antonio, TX, September 2007 (cited on p. 131).

[313] Q. Yang: „Recursive bilateral filtering“, In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid (Eds.): *Computer Vision – ECCV 2012, Part I*, Lecture Notes in Computer Science, Vol. 7572, 399–413, Springer, Berlin, 2012 (cited on pp. 168, 169).

[314] S. Yang and Y. Hu: „Coding artifact removal using biased anisotropic diffusion“, *Proc. 1997 IEEE International Conference on Image Processing*, Vol. 2, 346–349, Santa Barbara, CA, October 1997 (cited on p. 53).

[315] S. Yao, W. Lin, Z. Lu, E. Ong, and X. Yang: „Adaptive nonlinear diffusion processes for ringing artifacts removal on JPEG 2000 images“, *Proc. 2004 IEEE International Conference on Multimedia and Expo*, 691–694, Taipei, Taiwan, June 2004 (cited on p. 53).

[316] L. Yatziv and G. Sapiro: „Fast image and video colorization using chrominance blending“, *IEEE Transactions on Image Processing*, Vol. 15, No. 5, 1120–1129, May 2006 (cited on p. 169).

[317] T. Yoo and J. Coggins: „Using statistical pattern recognition techniques to control variable conductance diffusion“, In H. Barrett and A. Gmitro (Eds.): *Information Processing in Medical Imaging*, Lecture Notes in Computer Science, Vol. 687, 459–471, Springer, Berlin, 1993 (cited on p. 75).

[318] W. Yu, S. Heber, and T. Pock: „Learning reaction-diffusion models for image inpainting“, In J. Gall, P. Gehler, and B. Leibe (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 9358, 356–367, Springer, Berlin, 2015 (cited on pp. 130, 191).

[319] A. L. Yuille and T. A. Poggio: „Scaling theorems for zero crossings“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 1, 15–25, January 1986 (cited on p. 57).

[320] S. Zahir and M. Naqvi: „A near minimum sparse pattern coding based scheme for binary image compression“, *Proc. 2005 IEEE International Conference on Image Processing*, Vol. 2, II–289, Genova, Italy, September 2005 (cited on pp. 112, 115).

[321] G. Zeng and N. Ahmed: „A block coding technique for encoding sparse binary patterns“, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 37, No. 5, 778–780, May 1989 (cited on pp. 111, 112, 115, 116).

[322] C. Zhao and M. Du: „Image compression based on PDEs“, *Proc. 2011 International Conference of Computer Science and Network Technology*, Vol. 3, 1768–1771, Harbin, China, December 2011 (cited on pp. 55, 57).

[323] S. C. Zhu and D. Mumford: „Prior learning and Gibbs reaction-diffusion“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 11, 1236–1250, November 1997 (cited on pp. 69, 73, 74, 77, 79, 85).

[324] S. C. Zhu, Y. Wu, and D. Mumford: „Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling“, *International Journal of Computer Vision*, Vol. 27, No. 2, 107–126, March 1998 (cited on pp. 75, 85).

[325] H. Zimmer, L. Valgaerts, A. Bruhn, M. Breuß, J. Weickert, B. Rosenhahn, and H.-P. Seidel: „PDE-based anisotropic disparity-driven stereo vision“, In O. Deussen, D. Keim, and D. Saupe (Eds.): *Vision, Modelling, and Visualization 2008*, 263–272, AKA, Heidelberg, Germany, 2008 (cited on p. 69).

[326] H. Zimmer: „PDE-Based Image Compression Using Corner Information“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2007 (cited on p. 59).

[327] S. Zimmer: „Textures in PDE-based Image Compression“, Master Thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2013 (cited on p. 59).

[328] J. Ziv and A. Lempel: „An universal algorithm for sequential data compression“, *IEEE Transactions on Information Theory*, Vol. 23, No. 3, 337, May 1977 (cited on p. 49).

[329] J. Zujovic, T. Pappas, and D. Neuhoff: „Structural similarity metrics for texture analysis and retrieval“, *Proc. 16th IEEE International Conference on Image Processing*, 2225–2228, Cairo, Egypt, November 2009 (cited on p. 24).

Own Publications

B

> *An expert is a person who has found out by his own painful experience all the mistakes that one can make in a very narrow field.*
>
> — **Niels Bohr**
> (Physicist, Winner of the Nobel Prize)

1. P. Peter: „Three-dimensional data compression with anisotropic diffusion“, In J. Weickert, M. Hein, and B. Schiele (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8142, 231–236, Springer, Berlin, 2013.
2. P. Peter and M. Breuß: „Refined homotopic thinning algorithms and quality measures for skeletonisation methods“, In M. Breuß, A. Bruckstein, and P. Maragos (Eds.): *Innovations for Shape Analysis: Models and Algorithms*, Mathematics and Visualization, 77–92, Springer, Berlin, 2013.
3. P. Peter and J. Weickert: „Colour image compression with anisotropic diffusion“, *Proc. 21st IEEE International Conference on Image Processing*, 4822–4862, Paris, France, October 2014.
4. P. Peter and J. Weickert: „Compressing images with diffusion- and exemplar-based inpainting“, In J.-F. Aujol, M. Nikolova, and N. Papadakis (Eds.): *Scale-Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 9087, 154–165, Springer, Berlin, 2015.
5. P. Peter, C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert: „Beyond pure quality: Progressive modes, region of interest coding, and real time video decoding for PDE-based image compression“, *Journal of Visual Communication and Image Representation*, Vol. 31, No. 4, 253–265, August 2015.
6. P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert: *Evaluating the True Potential of Diffusion-Based Inpainting in a Compression Context*, Technical Report No. 373, Department of Mathematics, Saarland University, Saarbrücken, Germany, January 2016.
7. P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert: „From optimised inpainting with linear PDEs towards competitive image compression codecs“, To appear in T. Bräunl, B. McCane, M. Rivers, and X. Yu (Eds.): *Advances in Image and Video Technology*, Lecture Notes in Computer Science, Springer, Berlin, 2016.

8. P. Peter, J. Weickert, A. Munk, T. Krivobokova, and H. Li: „Justifying tensor-driven diffusion from structure-adaptive statistics of natural images“, In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 263–277, Springer, Berlin, 2015.
9. P. Peter, L. Kaufhold, and J. Weickert: *Turning Diffusion-based Image Colourisation into Efficient Colour Compression*, Technical Report No. 370, Department of Mathematics, Saarland University, Saarbrücken, Germany, December 2015.
10. 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*, Vol. 108, No. 3, 222–240, July 2014.

Glossary

C

AMLE Absolute Minimal Lipschitz Extension.

BMP Bitmap Image File.

BPG Better Portable Graphics.

BTTC B-tree Triangular Coding.

CCITT International Telegraph and Telephone Consultative Committee.

Convolution Integral transform that can be used to describe any liner shift invariant filter. In particular, linear homogeneous diffusion can be seen as Gaussian convolution. It is defined in Eq. 2.6.

DCT Discrete Cosine Transform.

DDD Directional Derivative Diffusion.

Diffusion Tensor Symmetric, positive matrix D that describes the directional behaviour of a diffusion process. Its eigenvalues indicate the amount of diffusion in the direction of the corresponding eigenvector.

DWT Discrete Wavelet Transform.

EED Edge-enhancing Anisotropic Diffusion.

EEDC Edge-enhancing Diffusion Codec.

FED Fast Explicit Diffusion.

FoE Field-of-Experts.

GIF Graphics Interchange Format.

GOP Group of Pictures.

GPU Graphics Processing Unit.

GRF Gibbs Random Field.

HFR High Frame Rate Video.

IEC International Electrotechnical Commission.

ISO International Organisation for Standardisation.

JPEG Joint Photographic Experts Group.

JPEG2000 Joint Photographic Experts Group 2000.

Lossless Compression Compression that removes redundancy from data and ensures an exact reconstruction.

Lossy Compression Compression that removes visually unimportant image data to increase the compression ratio.

LZW Lempel-Ziv-Welch Coding.

MRF Markov Random Field.

MSE Mean Squared Error.

NLI Nonlocal Inpainting.

NLNE Nonlocal Node Exchange.

NLPE Nonlocal Pixel Exchange.

PAQ Context-mixing scheme for compression of heterogeneous data. PAQ is not an acronym, but an onomatopoeic transcription of “pack”.

PCX Picture Exchange.

PDE Partial Differential Equation.

PNG Portable Network Graphics.

PPM Prediction by Partial Matching.

PSNR Peak Signal-to-noise Ratio.

Quantisation Discretisation of the co-domain of a continuous image. Grey or colour values are represented by a finite set of integer numbers instead of using a continuous range of real numbers.

R-EED Rectangular Subdivision with Edge-enhancing Diffusion.

RLE Run-length Encoding.

ROI Region of Interest Coding.

Sampling Discretisation of the spatial domain of a continuous image. Pixel locations live on a grid instead of a continuous coordinate system.

SOR Successive Overrelaxation.

SSIM Structural Similarity Measure.

Transform-based Compression Compression based on sparsity resulting from a transformation to another domain.

TV Total Variation.

VIF Visual Information Fidelity.

List of Symbols

D

a Global error threshold in R-EED, defined in Chapter 3.3.

$\boldsymbol{A}$ Semi-implicit discretisation of a diffusion PDE, defined in Eq. 3.48.

b Number of blocks in the hybrid texture compression algorithm of Chapter 6.

c Inpainting mask. Can be defined as a continuous confidence function $c : \Omega \rightarrow \mathbb{R}$, a continuous binary function, or a discrete vector $\boldsymbol{c}$ (see Chapter 3.1).

$\mathcal{C}$ Neighbourhood clique, defined in Chapter 4.

$c(P, \boldsymbol{f}, \boldsymbol{g})$ Contrast coefficient for the images $\boldsymbol{f}$ and $\boldsymbol{g}$ on a patch P, defined in Eq. 2.31.

c_P Confidence function for frame reuse in real-time video playback, defined in Eq. 7.13.

$\boldsymbol{D}$ Diffusion tensor, defined in its most general form in Eq. 4.36.

d Target mask density in stochast sparsification in Chapter 5. Also: Tree depth in R-EED, defined in Chapter 3.3.

$D^{\pm}_{x/y}$ Finite difference, defined in Eq. 3.40– Eq. 3.43.

div Divergence operator, defined in Eq. 2.11.

E Energy that is minimised in various variational approaches, see Chapter 4 and Chapter 6. Also: Edges of an MRF graph, defined in Chapter 4.

f Original image in an inpainting context, noisy initial image in a denoising context.

$G_{\mathcal{C}}$ Potential function for a neighbourhood clique $\mathcal{C}$, defined in Chapter 4.

g . . . . Diffusivity, determines amount of diffusion. Popular diffusivities are defined in Eq. 3.20.

$H(S)$ . . . . . . Entropy of an alphabet S, defined in Eq. 3.68.

h_P . . . . Homogeneity factor for confidence function in real-time video playback, defined in Eq. 7.12.

h_x, h_y . . . . . . Grid sizes in x- and y-direction for discrete images. See definition of sampling in Chapter 2.

$I(s)$ . . . . Self information of a symbol s, defined in Eq. 3.67.

$\boldsymbol{I}$ . . . . Unit matrix $\boldsymbol{I} :=$ diag$(1, \ldots, 1)$.

$\boldsymbol{J}$ . . . . Structure tensor, defined in Eq. 4.5.

$\boldsymbol{j}$ . . . . Flux according to Fick's law, defined in Eq. 3.1.

K . . . . Inpainting mask $K \subset \Omega$. Contains locations of known image data.

K_σ . . . . Gaussian with mean zero and standard deviation σ, defined in Eq. 2.7.

$\ell(P, \boldsymbol{f}, \boldsymbol{g})$ . . . . Mean luminance of the image $\boldsymbol{f}$ on a patch P, defined in Eq. 2.29.

ℓ . . . . Level factor in R-EED, defined in Chapter 3.3.

m . . . . Size of candidate set in stochastic sparsification or densification, see Chapter 5.

n . . . . Fraction of candidate points that are discarded again in stochastic sparsification or densification, see Chapter 5.

$\boldsymbol{n}$ . . . . Outer normal at image boundary $\partial\Omega$.

n_c . . . . Number of image channels (integer). Throughout this thesis, we only consider $n_c = 1$ for greyscale images and $n_c = 3$ for colour images.

$\mathcal{N}$ . . . . Neighbourhood system, defined in Chapter 4.

n_f . . . . Number of filters in a statistical prior (integer).

n_x, n_y . . . . . . Spatial image dimensions (integer).

$\mathcal{O}$ . . . . Bachmann-Landau notation, defined in Chapter 3.1.

p . . . . Probability density in Chapter 4. Also: Point pattern in R-EED, defined in Chapter 3.3.

P . . . . Image patch $P \subset \mathbb{R}$. Contains coordinates of a neighbourhood that defines a patch.

p_1, p_2, p_3 . . . . . Exponential SSIM weights, defined in Eq. 2.34.

$\mathcal{P}$ . . . . Power set.

q . . . . Number of quantised grey values.

$r_{i,j}$ . . . . Weights for region of interest coding in Chapter 8.1.

$\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g})$ . . . . . . Reconstruction from image data $\boldsymbol{g}$ and a binary mask $\boldsymbol{c}$ Chapter 5.

s_P . . . . Size factor for confidence function in real-time video playback, defined in Eq. 7.11.

$s(P, \boldsymbol{f}, \boldsymbol{g})$ . . . . Structure coefficient for the images $\boldsymbol{f}$ and $\boldsymbol{g}$ on a patch P, defined in Eq. 2.33.

T . . . . Subdivision tree in Chapter 5. Also: Adaptive error threshold in R-EED, defined in Chapter 3.3.

t . . . . Time variable.

t_1, t_2, t_3 . . . . . Thresholds for frame types in video encoding, see Chapter 7.

u . . . . Evolving image in a diffusion process.

V . . . . Vertices of an MRF graph, defined in Chapter 4. Also: Similarity function of non-local inpainting, defined in Eq. 6.1

$\boldsymbol{v}_i$ . . . . Eigenvector i of the structure or diffusion tensor.

w . . . . Gaussian-like weights of non-local inpainting, defined in Eq. 6.6.

x . . . . Spatial variable for x-direction (real-valued), image point $\boldsymbol{x}$ (vector-valued), or generic scalar.

y . . . . Spatial variable for y-direction (real-valued), image point $\boldsymbol{y}$ (vector-valued), or generic scalar.

α . . . . Discretisation weight in Chapter 3.1. Also: Luminance weight for SSIM, defined in Eq. 2.29.

α_k . . . . Free parameter for the k-th Student-t expert in a prior model, see Eq. 4.8.

β . . . . Discretisation weight in Chapter 3.1. Also: Contrast weight for SSIM, defined in Eq. 2.31.

χ_A . . . . Indicator function of the set A, defined in Eq. 3.27.

Δ . . . . Laplace operator, defined in Eq. 2.12.

Γ . . . . Discrete image domain $\Gamma \subset \mathbb{N}$. Contains indices $\{0, \ldots, n_x - 1\}$ and $\{0, \ldots, n_y - 1\}$ for x- and y-direction.

γ . . . . Discretisation weight in Chapter 3.1, diffusivity parameter in Chapter 4. Also: Contrast weight for SSIM, defined in Eq. 2.33.

λ . . . . Contrast parameter for diffusion models, see e.g. Eq. 3.20.

λ_i . . . . Eigenvalue i of the diffusion tensor.

μ . . . . Average grey value of an image, defined in Eq. 3.9.

μ_i . . . . Eigenvalue i of the structure tensor.

$\mu_{P,\boldsymbol{f}}$ . . . . Mean value of grey values of the image $\boldsymbol{f}$ on a patch P, defined in Eq. 2.28.

$\boldsymbol{\nabla}$ . . . . Gradient operator, defined in Eq. 2.10.

Ω . . . . Continuous image domain $\Omega \subset \mathbb{R}$. In this thesis, we assume that the domain is rectangular.

$\Omega \setminus K$ . . . . . . Inpainting domain $\Omega \setminus K \subset \Omega$. Contains locations of unknown image data.

ϕ . . . . Student-t expert function, see Eq. 4.8.

ψ . . . . Kurtotic distribution for variational penalisers, see Eq. 4.27. Also: Penaliser used to define a confidence function in Eq. 7.14

ρ . . . . Standard deviation of a Gaussian as defined in Eq. 2.7. In the context of diffusion, ρ describes the integration scale of the structure tensor (see Eq. 4.5).

σ . . . . Standard deviation of a Gaussian, defined in Eq. 2.7. In the context of diffusion, σ describes the smoothing scale of the structure tensor (see Eq. 4.5).

$\sigma_{P,\boldsymbol{f}}$ . . . . Standard deviation of an image $\boldsymbol{f}$ on a patch P, defined in Eq. 2.30.

$\sigma_{P,\boldsymbol{f},\boldsymbol{g}}$ . . . . . . Correlation coefficient of the images $\boldsymbol{f}$ and $\boldsymbol{g}$ on a patch P, defined in Eq. 2.32.

τ . . . . Discrete time step size, defined in Chapter 3.1.

$|\cdot|$ . . . . Absolute value or Euclidean norm (see Eq. 2.2).

$*$ . . . . Convolution operator, defined in Eq. 2.6.

$\lfloor\cdot\rfloor$ . . . . Floor operator, defined in Eq. 2.8.

$\|$ . . . . parallel, i.e. for two vectors $\boldsymbol{u} \parallel \boldsymbol{v} \Leftrightarrow \boldsymbol{u} = a\boldsymbol{v}$ with $a \in \mathbb{R}$.

∂A . . . . Boundary of a set A.

$\partial_{\boldsymbol{v}}$ . . . . Directional derivative in respect to the vectorial direction $\boldsymbol{v} \in \mathbb{R}^n$, defined in Eq. 2.13.

d_u . . . . Gâteaux derivative in respect to a function $\boldsymbol{u} : \mathbb{R} \mapsto \mathbb{R}$, defined in Eq. 2.15.

∂_x . . . . Partial derivative in respect to argument $x \in \mathbb{R}$, defined in Eq. 2.9.

$\perp$ . . . . perpendicular/orthogonal, defined in Eq. 2.4.

List of Figures

List of Tables

G Detailed Derivations for Probabilistic Diffusion Models

> *Computations are everywhere, once you begin to look at things in a certain way.*
>
> — **Rudy von Bitter Rucker**
> (Mathematician and Science Fiction Author)

In this appendix we cover the computations necessary to derive the newly introduced PDEs in Chapter 4 in more detail. First, we derive the PDE corresponding to the unifying probabilistic framework from Chapter 4.5 in Section G.1. The EED-related DDD model from Chapter 4.8 is the topic of Section G.2.

In both cases, we are interested in the necessary conditions, that a minimiser $\boldsymbol{u} : \mathbb{R}^2 \mapsto \mathbb{R}^{n_c}$ of an energy

$$E(\boldsymbol{u}) = \int_\Omega F(x, y, \boldsymbol{u})\, dx\, dy \tag{G.1}$$

has to fulfil. To this end, we compute the Gâteaux-derivatives

$$\frac{d}{d\epsilon} \int_\Omega F(x, y, \boldsymbol{u} + \mathbf{diag}(\boldsymbol{h})\boldsymbol{\epsilon})\, dx\, dy \bigg|_{\epsilon=0} . \tag{G.2}$$

for $k = 1, \ldots, n_c$, the test function $\boldsymbol{h} : \mathbb{R}^2 \mapsto \mathbb{R}^{n_x n_y}$, and $\boldsymbol{\epsilon} \in \mathbb{R}^{n_c}$. The term $\mathbf{diag}(\boldsymbol{h})$ denotes an $n_c \times n_c$ diagonal matrix that contains the components of the vector-valued test function $\boldsymbol{h}$ on its diagonal:

$$\mathbf{diag}(\boldsymbol{h}) := \begin{pmatrix} h_1 & 0 & 0 & 0 \\ 0 & h_2 & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & h_{n_c} \end{pmatrix} \tag{G.3}$$

These derivations follow well-known rules from variational calculus. We have collected some of the relations that are relevant for both cases in Section G.3 to reduce overhead.

Note that the treatment of the structure tensor derivatives in Section G.1 follows the formulation of Roussos and Maragos [239], who derive a special case of our general minimiser in their paper.

G.1 Deriving the Unifying Probabilistic Diffusion Framework

In Chapter 4, we want to derive diffusion PDEs from the unifying energy

$$E(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left(\frac{1}{\tau}|\boldsymbol{u}-\boldsymbol{f}|^2 + \psi(\mu_1,\mu_2)\right) dx\, dy \tag{G.4}$$

that contains the noise prior $\tau|\boldsymbol{u}-\boldsymbol{f}|$ and the natural image prior $\psi(\mu_1,\mu_2)$. The latter is a function of the eigenvalues μ_1 and μ_2 of the structure tensor

$$\boldsymbol{J}_{n_c,\rho,\sigma} := K_\rho * \left(\sum_{k=1}^{m} \boldsymbol{\nabla}_\sigma\, u_k \boldsymbol{\nabla}_\sigma u_k^\top\right) \tag{G.5}$$

The noise prior directly leads to a term that can be interpreted as a discrete time derivative with step size τ. However, we still need to compute the divergence term from the Gâteaux-derivative of the image prior

$$0 = \frac{1}{\tau}(u_k - f_k) + \frac{1}{2}\int_\Omega \Big(d_{\epsilon_k}(\psi(\mu_1(\boldsymbol{u}+\boldsymbol{H\epsilon}),\ \mu_2(\boldsymbol{u}+\boldsymbol{H\epsilon})))\Big)\, dx\, dy \tag{G.6}$$

for $k \in \{1,...,n_c\}$. As already mentioned in Chapter 4, we focus on the derivatives

$$d_{\epsilon_k}(\psi(\mu_1(\boldsymbol{u}+\boldsymbol{H\epsilon}),\ \mu_2(\boldsymbol{u}+\boldsymbol{H\epsilon}))) \;=\; \frac{\partial\psi}{\partial\mu_1} d_{\epsilon_k}(\mu_1) \;+\; \frac{\partial\psi}{\partial\mu_2} d_{\epsilon_k}(\mu_2) \tag{G.7}$$

of the penaliser ψ, first.

Derivatives of Structure Tensor Entries

The eigenvalues of the structure tensor $\boldsymbol{J}_{n_c,\sigma,\rho}$ can be defined in terms of its matrix elements $J_{i,j},\ i,j \in \{1,2\}$. In order to compute the Gâteaux-derivatives of μ_1 and μ_2, we therefore establish a common structure for the derivatives of $J_{i,j}$ first. This simplifies later computations significantly.

$$\begin{aligned}
\frac{d}{d\epsilon_k} J_{11}\Big|_{\epsilon_k=0} &= K_\rho * 2\partial_x u_{k,\sigma} \partial_x h_{k,\sigma}\,, \\
\frac{d}{d\epsilon_k} J_{22}\Big|_{\epsilon_k=0} &= K_\rho * 2\partial_y u_{k,\sigma} \partial_y h_{k,\sigma}\,, \\
\frac{d}{d\epsilon_k} J_{12}\Big|_{\epsilon_k=0} &= K_\rho * \partial_x u_{k,\sigma} \partial_y h_{k,\sigma} + \partial_y u_{k,\sigma} \partial_x h_{k,\sigma}\,, \\
&= K_\rho * (\partial_y u_{k,\sigma}, \partial_x u_{k,\sigma}) \boldsymbol{\nabla} h_{k,\sigma}\,.
\end{aligned} \tag{G.8}$$

The derivative $\frac{d}{d\epsilon_k} J_{12}|_{\epsilon_k=0}$ allows a direct formulation as the dot product between $\boldsymbol{\nabla} h_{k,\sigma}$ and $(\partial_y u_{k,\sigma}, \partial_x u_{k,\sigma})^\top$. For the components J_{11} and J_{22}, we cannot achieve such a structure immediately. However, in later computations we will only need sums and differences of J_{11} and J_{22}. To this end, we can simplify both terms as dot products:

$$\begin{aligned} \frac{d}{d\epsilon_k}(J_{11}+J_{22})\bigg|_{\epsilon_k=0} &= K_\rho * 2\Big(\partial_x u_{k,\sigma}\partial_x h_{k,\sigma} + \partial_y u_{k,\sigma}\partial_y h_{k,\sigma}\Big) \\ &= K_\rho * 2\boldsymbol{\nabla} u_{k,\sigma}^\top \boldsymbol{\nabla} h_{k,\sigma}\,, \\ \frac{d}{d\epsilon_k}(J_{11}-J_{22})\bigg|_{\epsilon_k=0} &= K_\rho * 2\Big(\partial_x u_{k,\sigma}\partial_x h_{k,\sigma} - \partial_y u_{k,\sigma}\partial_y h_{k,\sigma}\Big) \\ &= K_\rho * 2(\partial_x u_{k,\sigma}, -\partial_y u_{k,\sigma})\boldsymbol{\nabla} h_{k,\sigma}\,. \end{aligned} \tag{G.9}$$

Note that for the special case $\sigma = 0$ and with a different notation ($\boldsymbol{M}_1 \boldsymbol{\nabla} u = \boldsymbol{\nabla} u_\perp$, $\boldsymbol{M}_2 \boldsymbol{\nabla} u = (u_y, u_x)^\top$ with suitable matrices $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$) these derivatives are consistent with the intermediate results of Roussos and Maragos in Section 6.2 of their publication [239].

Derivatives of the Eigenvalues

Now that we have a compact notation for the derivatives of the structure tensor entries, we can deal with the derivatives of the eigenvalues μ_1 and μ_2. In order to shorten notations, let $\boldsymbol{v} := (\partial_x u_{k,\sigma}, -\partial_y u_{k,\sigma})$ and $\boldsymbol{w} := (\partial_y u_{k,\sigma}, \partial_x u_{k,\sigma})$.

$$\begin{aligned} \frac{d}{d\epsilon_k}\mu_1\bigg|_{\epsilon_k=0} &= \frac{d}{d\epsilon_k}\frac{1}{2}\Big(J_{11}+J_{22}+\sqrt{(J_{11}-J_{22})^2+4J_{12}^2}\Big)\bigg|_{\epsilon_k=0} \\ &= \frac{\frac{d}{d\epsilon_k}J_{11}+J_{22}|_{\epsilon_k=0}}{2} + \frac{(J_{11}-J_{22})\frac{d}{d\epsilon_k}(J_{11}-J_{22})|_{\epsilon_k=0} + 4J_{12}\frac{d}{d\epsilon_k}J_{12}|_{\epsilon_k=0}}{2\sqrt{(J_{11}-J_{22})^2+4J_{12}^2}} \\ &= K_\rho * \left(\left(\boldsymbol{\nabla} u_{k,\sigma}^\top + \frac{(J_{11}-J_{22})\boldsymbol{v} + 2J_{12}\boldsymbol{w}}{\sqrt{(J_{11}-J_{22})^2+4J_{12}^2}}\right)\boldsymbol{\nabla} h_{k,\sigma}\right) \\ &= K_\rho * \Big((\boldsymbol{M}\boldsymbol{\nabla} u_{k,\sigma})^\top \boldsymbol{\nabla} h_{k,\sigma}\Big) \end{aligned} \tag{G.10}$$

Note that $\sqrt{(J_{11}-J_{22})^2+4J_{12}^2} = \mu_1 - \mu_2$ which enables us to define the matrix $\boldsymbol{M}$ as

$$\boldsymbol{M} := \frac{1}{\mu_1-\mu_2}\begin{pmatrix} \mu_1-\mu_2+J_{11}-J_{22} & 2J_{12} \\ 2J_{12} & \mu_1-\mu_2-J_{11}+J_{22} \end{pmatrix}. \tag{G.11}$$

The matrix $\boldsymbol{M}$ can be directly related to the eigenvectors of $\boldsymbol{J}_{n_c,\sigma,\rho}$ in a similar way as in [239] by computing the eigenvalues λ_1 and λ_2 and the first eigenvector $\boldsymbol{u}_1$ of $\boldsymbol{M}$ directly:

$$\begin{aligned}
\lambda_1 &= \frac{M_{11} + M_{22} + \sqrt{(M_{11} - M_{22})^2 + 4M_{12}^2}}{2} \\
&= \frac{2 + \sqrt{4\frac{(J_{11}-J_{22})^2+4J_{12}^2}{(J_{11}-J_{22})^2+4J_{12}^2}}}{2} = \frac{2+\sqrt{4}}{2} = 2\,, \\
\lambda_2 &= \frac{2-\sqrt{4}}{2} = 0\,, \\
\hat{\boldsymbol{u}}_1 &= \begin{pmatrix} M_{11} - M_{22} + \sqrt{(M_{11} - M_{22})^2 + 4M_{12}^2} \\ 2M_{12} \end{pmatrix} \\
&= \frac{2}{\lambda_1 - \lambda_2} \begin{pmatrix} J_{11} - J_{22} + \sqrt{(J_{11} - J_{22})^2 + 4J_{12}^2} \\ 2J_{12} \end{pmatrix}\,.
\end{aligned} \tag{G.12}$$

Thus, the eigenvector corresponding to the only non-zero eigenvalue of $\boldsymbol{M}$ has the same direction as $\hat{\boldsymbol{u}}_1$. This vector is parallel to $\boldsymbol{v}_1$, i.e. after normalisation $\boldsymbol{u}_1 = \hat{\boldsymbol{u}}_1 / \|\hat{\boldsymbol{u}}_1\| = \boldsymbol{v}_1$. From this and the fact that $\lambda_1 = 2$ and $\lambda_2 = 0$ we can conclude that

$$\boldsymbol{M} = 2\boldsymbol{v}_1\boldsymbol{v}_1^\top\,. \tag{G.13}$$

With analogous computations for $\frac{d}{d\epsilon_k}\mu_2|_{\epsilon_k=0}$ we finally arrive at the following compact notation for the Gâteaux-derivatives of the eigenvalues:

$$\begin{aligned}
\frac{d}{d\epsilon_k}\mu_1\bigg|_{\epsilon_k=0} &= K_\rho * \left((2\boldsymbol{v}_1\boldsymbol{v}_1^\top \boldsymbol{\nabla} u_{k,\sigma})^\top \boldsymbol{\nabla} h_{k,\sigma}\right)\,, \\
\frac{d}{d\epsilon_k}\mu_2\bigg|_{\epsilon_k=0} &= K_\rho * \left((2\boldsymbol{v}_2\boldsymbol{v}_2^\top \boldsymbol{\nabla} u_{k,\sigma})^\top \boldsymbol{\nabla} h_{k,\sigma}\right)\,.
\end{aligned} \tag{G.14}$$

Divergence Term

With the results from Eq. G.14 we can finally compute the missing divergence term in Eq. G.6. We exploit the fact that integral

$$\frac{1}{2}\int_\Omega \left(\frac{\partial}{\partial\mu_1}\psi(\mu_1,\mu_2)\frac{d}{d\epsilon_k}\mu_1\bigg|_{\epsilon_k=0} + \frac{\partial}{\partial\mu_2}\psi(\mu_1,\mu_2)\frac{d}{d\epsilon_k}\mu_2\bigg|_{\epsilon_k=0}\right) dx\,dy \tag{G.15}$$

can be split into two parts. First, we explicitly derive the Gâteaux-Derivative for the part related to the first eigenvalue μ_1.

$$\begin{aligned}
&\frac{1}{2}\int_\Omega \partial\mu_1\psi \cdot K_\rho * \Big((2\boldsymbol{v}_1\boldsymbol{v}_1^\top \boldsymbol{\nabla} u_{k,\sigma})^\top \boldsymbol{\nabla} h_{k,\sigma}\Big)\,dx\,dy \\
\overset{\text{(G.35)}}{=}\; &\int_\Omega (K_\rho * \partial\mu_1\psi)\Big((\boldsymbol{v}_1\boldsymbol{v}_1^\top \boldsymbol{\nabla} u_{k,\sigma})^\top \boldsymbol{\nabla} h_{k,\sigma}\Big)\,dx\,dy \\
=\; &\int_\Omega \Big((K_\rho * (\partial\mu_1\psi)\boldsymbol{v}_1\boldsymbol{v}_1^\top)\boldsymbol{\nabla} u_{k,\sigma}\Big)^\top \boldsymbol{\nabla} h_{k,\sigma}\,dx\,dy \\
\overset{\text{(G.37)}}{=}\; &\int_\Omega -K_\sigma * \operatorname{div}\Big((K_\rho * (\partial\mu_1\psi)\boldsymbol{v}_1\boldsymbol{v}_1^\top)\boldsymbol{\nabla} u_{k,\sigma}\Big) h_{k,\sigma}\,dx\,dy
\end{aligned} \tag{G.16}$$

The computations above rely on rules for variational calculus with Guassian convolutions from Section G.3. A similar treatment of the second part and a combination of both results leads to the final Gâteaux-derivative

$$\begin{aligned}
&\frac{1}{2}\int_\Omega \Big(d_{\epsilon_k}(\psi(\mu_1(\boldsymbol{u}+\boldsymbol{H\epsilon}),\ \mu_2(\boldsymbol{u}+\boldsymbol{H\epsilon})))\Big)\,dx\,dy \\
=&\int_\Omega -K_\sigma * \operatorname{div}\Big((K_\rho * \boldsymbol{D})\boldsymbol{\nabla} u_{k,\sigma}\Big) h_{k,\sigma}\,dx\,dy
\end{aligned} \tag{G.17}$$

with the diffusion tensor

$$\boldsymbol{D} = \partial_{\mu_1}\psi(\mu_1,\mu_2)\boldsymbol{v}_1\boldsymbol{v}_1^\top + \partial_{\mu_2}\psi(\mu_1,\mu_2)\boldsymbol{v}_2\boldsymbol{v}_2^\top. \tag{G.18}$$

Plugging the results of Eq. G.17 into Eq. G.6 finally yields

$$\frac{u_k - f_k}{\tau} \;=\; \boldsymbol{\nabla}_\sigma^\top((K_\rho{*}\boldsymbol{D})\;\boldsymbol{\nabla}_\sigma u_k), \qquad k = 1, ..., n_c. \tag{G.19}$$

This concludes the detailed version of our proof.

G.2 Deriving the Directional Diffusion Model

In this Section, we consider an energy that relies on the prior functions $\psi_1, \psi_2 : \mathbb{R} \mapsto \mathbb{R}$ that represent the distribution of the of directional derivatives along the presmoothed gradient $\boldsymbol{\nabla} u_\sigma$ and the orthogonal direction $\boldsymbol{\nabla} u_\sigma{}^\perp$. To obtain a compact notation we define

$$\partial_{\boldsymbol{\nabla} u_\sigma} u := \frac{\boldsymbol{\nabla} u_\sigma^\top}{|\boldsymbol{\nabla} u_\sigma|} \boldsymbol{\nabla} u, \tag{G.20}$$

$$\partial_{\boldsymbol{\nabla} u_\sigma^\perp} u := \frac{\boldsymbol{\nabla} u_\sigma^{\perp\,\top}}{|\boldsymbol{\nabla} u_\sigma|} \boldsymbol{\nabla} u. \tag{G.21}$$

Now, we consider the energy

$$E(u) = \frac{1}{2} \int_\Omega \underbrace{\psi_1\left(\left(\partial_{\boldsymbol{\nabla} u_\sigma} u\right)^2\right) + \psi_2\left(\left(\partial_{\boldsymbol{\nabla} u_\sigma^\perp} u\right)^2\right)}_{=:F(x,y,u)} \, dx\, dy \tag{G.22}$$

and want to compute the corresponding Gâteaux-derivative

$$\frac{d}{d\epsilon} \frac{1}{2} \int_\Omega F(x, y, u + \epsilon h)\, dx\, dy \bigg|_{\epsilon=0} . \tag{G.23}$$

Preparatory Derivative Calculations

In the same way as for the unifying model, we first compute some derivatives that we can use as building blocks for a compact notation of our final Gâteaux-derivatives. We start with the squared magnitude of the smoothed gradient which requires the normalisation of the directional derivatives:

$$\begin{aligned}
\frac{d}{d\epsilon}\left(|\boldsymbol{\nabla}(u+\epsilon h)_\sigma|^2\right)\bigg|_{\epsilon=0} &= \frac{d}{d\epsilon}\left((\partial_x u_\sigma + \epsilon \partial_x h_\sigma)^2 + (\partial_y u_\sigma + \epsilon \partial_y h_\sigma)^2\right)\bigg|_{\epsilon=0} \\
&= \Big(2(\partial_x (u+\epsilon h)_\sigma \partial_x h_\sigma) + 2(\partial_y (u+\epsilon h)_\sigma \partial_y h_\sigma)\Big)\bigg|_{\epsilon=0} \\
&= 2 \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} h_\sigma\,.
\end{aligned} \tag{G.24}$$

In addition, we need the dot product between the smoothed and unsmoothed gradient, which amounts to

$$\begin{aligned}
&\frac{d}{d\epsilon}\left(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \boldsymbol{\nabla}(u+\epsilon h)\right)\bigg|_{\epsilon=0} \\
&= \frac{d}{d\epsilon} \boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \bigg|_{\epsilon=0} \boldsymbol{\nabla} u + \boldsymbol{\nabla} u_\sigma^\top \frac{d}{d\epsilon} \boldsymbol{\nabla}(u+\epsilon h)\bigg|_{\epsilon=0} \\
&= \boldsymbol{\nabla} h_\sigma^\top \boldsymbol{\nabla} u + \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} h.
\end{aligned} \tag{G.25}$$

From there on, we can proceed to the last building block that we need, the squared dot product

$$\begin{aligned}
&\frac{d}{d\epsilon}(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \boldsymbol{\nabla}(u+\epsilon h))^2\Big|_{\epsilon=0} \\
=\;& 2(\boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)\frac{d}{d\epsilon}\left(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \boldsymbol{\nabla}(u+\epsilon h)\right)\Big|_{\epsilon=0} \\
\overset{(\text{G.25})}{=}\;& 2(\boldsymbol{\nabla} u \boldsymbol{\nabla} u^\top \boldsymbol{\nabla} u_\sigma)^\top \boldsymbol{\nabla} h_\sigma + 2(\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)^\top \boldsymbol{\nabla} h\,.
\end{aligned} \tag{G.26}$$

Finally, we can compute the Gâteaux-derivative of the full argument of ψ_1, which leads to

$$\begin{aligned}
&\frac{d}{d\epsilon}\left(\frac{(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \boldsymbol{\nabla}(u+\epsilon h))^2}{|\boldsymbol{\nabla}(u+\epsilon h)_\sigma|^2}\right)\Big|_{\epsilon=0} \\
=\;& \frac{|\boldsymbol{\nabla} u_\sigma|^2 \frac{d}{d\epsilon}(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top \boldsymbol{\nabla}(u+\epsilon h))^2\Big|_{\epsilon=0} - \frac{d}{d\epsilon}\left(|\boldsymbol{\nabla}(u+\epsilon h)_\sigma|^2\right)\Big|_{\epsilon=0}(\boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)^2}{|\boldsymbol{\nabla} u_\sigma|^4} \\
\underset{(\text{G.26})}{\overset{(\text{G.24})}{=}}\;& \frac{2(\boldsymbol{\nabla} u \boldsymbol{\nabla} u^\top \boldsymbol{\nabla} u_\sigma)^\top \boldsymbol{\nabla} h_\sigma}{|\boldsymbol{\nabla} u_\sigma|^2} + \frac{2(\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)^\top \boldsymbol{\nabla} h}{|\boldsymbol{\nabla} u_\sigma|^2} - \frac{2(\boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)^2 \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} h_\sigma}{|\boldsymbol{\nabla} u_\sigma|^4} \\
=\;& \frac{2\left(\left(\boldsymbol{\nabla} u \boldsymbol{\nabla} u_\sigma^\top - \frac{(\boldsymbol{\nabla} u_\sigma{}^\top \boldsymbol{\nabla} u)\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma{}^\top}{|\boldsymbol{\nabla} u_\sigma|^2}\right)\boldsymbol{\nabla} u\right)^\top \boldsymbol{\nabla} h_\sigma}{|\boldsymbol{\nabla} u_\sigma|^2} + \frac{2(\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\top \boldsymbol{\nabla} u)^\top \boldsymbol{\nabla} h}{|\boldsymbol{\nabla} u_\sigma|^2}.
\end{aligned} \tag{G.27}$$

Repeating four analogous steps for the orthogonal direction $\boldsymbol{\nabla} u_\sigma{}^\perp$ yields

$$\begin{aligned}
&\frac{d}{d\epsilon}\left(\frac{(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\perp{}^\top \boldsymbol{\nabla}(u+\epsilon h))^2}{|\boldsymbol{\nabla}(u+\epsilon h)_\sigma|^2}\right)\Big|_{\epsilon=0} \\
=\;& \frac{2\left(\left(-\boldsymbol{\nabla} u^\perp \boldsymbol{\nabla} u_\sigma^\perp{}^\top - \frac{(\boldsymbol{\nabla} u_\sigma^\perp{}^\top \boldsymbol{\nabla} u)\boldsymbol{\nabla} u_\sigma \boldsymbol{\nabla} u_\sigma^\perp{}^\top}{|\boldsymbol{\nabla} u_\sigma|^2}\right)\boldsymbol{\nabla} u\right)^\top \boldsymbol{\nabla} h_\sigma}{|\boldsymbol{\nabla} u_\sigma|^2} + \frac{2(\boldsymbol{\nabla} u_\sigma^\perp u_\sigma^\perp{}^\top \boldsymbol{\nabla} u)^\top \boldsymbol{\nabla} h}{|\boldsymbol{\nabla} u_\sigma|^2}.
\end{aligned} \tag{G.28}$$

Note that we have exploited here the identity $\boldsymbol{\nabla} u^\top \boldsymbol{\nabla} h_\sigma^\perp = -\boldsymbol{\nabla} u^\perp{}^\top \boldsymbol{\nabla} h_\sigma$ to obtain the same structure as in Eq. G.27. In the following, we combine the results from Eq. G.27 and Eq. G.28 to obtain the Euler-Lagrange equation.

Euler-Lagrange Equation

We split our energy again in two separate derivatives, one for ψ_1, and one for ψ_2:

$$\begin{aligned}
E(u) = \frac{1}{2}\int_\Omega \frac{d}{d\epsilon}\psi_1\left((\partial_{\boldsymbol{\nabla}(u+\epsilon h)_\sigma}(u+\epsilon h))^2\right)\Big|_{\epsilon=0} dx\,dy \\
+\frac{1}{2}\int_\Omega \frac{d}{d\epsilon}\psi_2\left((\partial^\perp_{\boldsymbol{\nabla}(u+\epsilon h)_\sigma}(u+\epsilon h))^2\right)\Big|_{\epsilon=0} dx\,dy\,.
\end{aligned} \tag{G.29}$$

Now, we can use Eq. G.27 to compute the first partial problem and obtain

$$
\begin{aligned}
&\frac{1}{2}\int_\Omega \frac{d}{d\epsilon}\psi_1\left((\partial_{\boldsymbol{\nabla}(u+\epsilon h)_\sigma}(u+\epsilon h))^2\right)\Big|_{\epsilon=0} dx\,dy \\
=\quad & \int_\Omega \left(\psi_1'\left((\partial_{\boldsymbol{\nabla}(u+\epsilon h)_\sigma}(u+\epsilon h))^2\right)\frac{d}{d\epsilon}\left(\frac{(\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\top\boldsymbol{\nabla}(u+\epsilon h))^2}{|\boldsymbol{\nabla}(u+\epsilon h)_\sigma|^2}\right)\right)\Big|_{\epsilon=0} dx\,dy \\
\overset{\text{(G.27)}}{=}\quad & \int_\Omega \psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{\left(\left(\boldsymbol{\nabla}u\boldsymbol{\nabla}u_\sigma^\top-\frac{(\boldsymbol{\nabla}u_\sigma{}^\top\boldsymbol{\nabla}u)\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma{}^\top}{|\boldsymbol{\nabla}u_\sigma|^2}\right)\boldsymbol{\nabla}u\right)^\top\boldsymbol{\nabla}h_\sigma}{|\boldsymbol{\nabla}u_\sigma|^2}\,dx\,dy \\
&+\int_\Omega \psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{(\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma^\top\boldsymbol{\nabla}u)^\top\boldsymbol{\nabla}h}{|\boldsymbol{\nabla}u_\sigma|^2}\,dx\,dy \\
\underset{\text{(G.37)}}{\overset{\text{(G.36)}}{=}}\quad & \int_\Omega -\operatorname{div}\left(K_\sigma*\left(\psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{\left(\boldsymbol{\nabla}u\boldsymbol{\nabla}u_\sigma^\top-\frac{\partial_{\boldsymbol{\nabla}u_\sigma}u\,\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma{}^\top}{|\boldsymbol{\nabla}u_\sigma|}\right)\boldsymbol{\nabla}u}{|\boldsymbol{\nabla}u_\sigma|^2}\right)\right)h\,dx\,dy \\
&+\int_\Omega -\operatorname{div}\left(\psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma^\top\boldsymbol{\nabla}u}{|\boldsymbol{\nabla}u_\sigma|^2}\right)h\,dx\,dy\,.
\end{aligned}
\tag{G.30}
$$

In the same way, we use Eq. G.28 to compute

$$
\begin{aligned}
&\frac{1}{2}\int_\Omega \frac{d}{d\epsilon}\psi_2\left((\partial_{\boldsymbol{\nabla}(u+\epsilon h)_\sigma^\perp}(u+\epsilon h))^2\right)\Big|_{\epsilon=0} dx\,dy \\
=\quad & \int_\Omega -\operatorname{div}\left(K_\sigma*\left(\psi_2'\left((\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u)^2\right)\frac{-\boldsymbol{\nabla}u^\perp\boldsymbol{\nabla}u_\sigma^\perp{}^\top-\frac{\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u\,\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma^\perp{}^\top}{|\boldsymbol{\nabla}u_\sigma|}\boldsymbol{\nabla}u}{|\boldsymbol{\nabla}u_\sigma|^2}\right)\right)h\,dx\,dy \\
&+\int_\Omega -\operatorname{div}\left(\psi_2'\left((\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u)^2\right)\frac{\boldsymbol{\nabla}u_\sigma^\perp\boldsymbol{\nabla}u_\sigma^\perp{}^\top\boldsymbol{\nabla}u}{|\boldsymbol{\nabla}u_\sigma|^2}\right)h\,dx\,dy.
\end{aligned}
\tag{G.31}
$$

Finally, the Euler-Lagrange equation is

$$
\begin{aligned}
0\;=\;&-\operatorname{div}\Bigg(\Bigg((\psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma^\top}{|\boldsymbol{\nabla}u_\sigma|^2}+\psi_2'\left((\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u)^2\right)\frac{\boldsymbol{\nabla}u_\sigma^\perp\boldsymbol{\nabla}u_\sigma^\perp{}^\top}{|\boldsymbol{\nabla}u_\sigma|^2}\Bigg)\boldsymbol{\nabla}u\Bigg) \\
&-\operatorname{div}\Bigg(K_\sigma*\Bigg(\Bigg(\psi_1'\left((\partial_{\boldsymbol{\nabla}u_\sigma}u)^2\right)\frac{\boldsymbol{\nabla}u\boldsymbol{\nabla}u_\sigma^\top-\frac{\partial_{\boldsymbol{\nabla}u_\sigma}u\,\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma{}^\top}{|\boldsymbol{\nabla}u_\sigma|}}{|\boldsymbol{\nabla}u_\sigma|^2} \\
&\qquad\qquad+\psi_2'\left((\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u)^2\right)\frac{-\boldsymbol{\nabla}u^\perp\boldsymbol{\nabla}u_\sigma^\perp{}^\top-\frac{\partial_{\boldsymbol{\nabla}u_\sigma^\perp}u\,\boldsymbol{\nabla}u_\sigma\boldsymbol{\nabla}u_\sigma^\perp{}^\top}{|\boldsymbol{\nabla}u_\sigma|}}{|\boldsymbol{\nabla}u_\sigma|^2}\Bigg)\boldsymbol{\nabla}u\Bigg)\Bigg)\,.
\end{aligned}
\tag{G.32}
$$

This concludes our detailed computation since we have arrived at the formulation of Chapter 4.8: We have one divergence term that resembles EED and an additional anisotropic divergence term.

Reduction to Isotropic Diffusion

For $\sigma = 0$, the equation can be significantly simplified:

$$\begin{aligned} 0 = &- \operatorname{div}\Bigg((\psi_1'(|\boldsymbol{\nabla} u|^2) \underbrace{\frac{\boldsymbol{\nabla} u^\top \boldsymbol{\nabla} u}{|\boldsymbol{\nabla} u|^2}}_{=1} \boldsymbol{\nabla} u + \psi_2'(0) \underbrace{\frac{{\boldsymbol{\nabla} u^\perp}^\top \boldsymbol{\nabla} u}{|\boldsymbol{\nabla} u|^2}}_{=0} \boldsymbol{\nabla} u^\perp \Bigg) \\ &- \operatorname{div}\Bigg(K_\sigma * \Bigg(\Bigg(\psi_1' \left((\partial_{\boldsymbol{\nabla} u} u)^2 \right) \underbrace{\left(1 - \frac{(\partial_{\boldsymbol{\nabla} u} u)}{|\boldsymbol{\nabla} u|} \right)}_{=0} \boldsymbol{\nabla} u \Bigg) \frac{\boldsymbol{\nabla} u^\top \boldsymbol{\nabla} u}{|\boldsymbol{\nabla} u|^2} \\ &\qquad + \psi_2' \left((\partial_{\boldsymbol{\nabla} u^\perp} u)^2 \right) \frac{-\boldsymbol{\nabla} u^\perp - \frac{(\partial_{\boldsymbol{\nabla} u^\perp} u) \boldsymbol{\nabla} u}{|\boldsymbol{\nabla} u|}}{|\boldsymbol{\nabla} u|^2} \underbrace{{\boldsymbol{\nabla} u^\perp}^\top \boldsymbol{\nabla} u}_{=0} \Bigg) \Bigg) , \end{aligned} \tag{G.33}$$

and one obtains the isotropic case

$$0 = -\operatorname{div} \left(\psi_1'(|\boldsymbol{\nabla} u|^2) \boldsymbol{\nabla} u \right) . \tag{G.34}$$

G.3 Common Relations for Variational Derivations

In order to compute the minimisers in the previous sections, we need some well-known results from variational calculus. For $F : \mathbb{R}^2 \mapsto \mathbb{R}$, $\boldsymbol{f} : \mathbb{R}^2 \mapsto \mathbb{R}^2$, a test function $h : \mathbb{R}^2 \mapsto \mathbb{R}$, and a symmetric convolution kernel K with $K(x, y) = K(-x, -y)$, we have:

$$\begin{aligned} \int_\Omega F \cdot (K * h) \, dx \, dy &= \int_\Omega F(x, y) \left(\int_\Omega K(x - u, y - v) \, h(u, v) \, du \, dv \right) dx \, dy \qquad \text{(G.35)} \\ &= \int_\Omega \left(\int_\Omega F(x, y) \underbrace{K(x - u, y - v)}_{= K(u - x, v - y)} \, dx \, dy \right) h(u, v) \, du \, dv \\ &= \int_\Omega (K * F) \, h \, du \, dv \, , \\ \int_\Omega \boldsymbol{f}^\top \boldsymbol{\nabla} h \, dx \, dy &= \int_\Omega f_1 \, h_x + f_2 \, h_y \, dx \, dy \qquad \text{(G.36)} \\ &= - \int_\Omega \partial_x f_1 \, h \, dx \, dy - \int_\Omega \partial_y f_2 \, h \, dx \, dy \\ &= \int_\Omega -\operatorname{div}(\boldsymbol{f}) \, h \, dx \, dy \, , \\ \int_\Omega \boldsymbol{f}^\top \boldsymbol{\nabla} (K * h) \, dx \, dy &= \int_\Omega f_1 \, (K * h_x) + f_2 \, (K * h_y) \, dx \, dy \qquad \text{(G.37)} \\ &\overset{\text{(G.35)}}{=} \int_\Omega (K * f_1) \, h_x + (K * f_2) \, h_y \, dx \, dy \\ &\overset{\text{(G.36)}}{=} \int_\Omega -\operatorname{div}(K * \boldsymbol{f}) \, h \, dx \, dy \, . \end{aligned}$$

H Implementations

> *Normal people don't understand this concept;*
> *they believe that if it ain't broke, don't fix it.*
> *Engineers believe that if it ain't broke,*
> *it doesn't have enough features yet.*
>
> — **Scott Adams**
> (Cartoonist)

Most of the program code for this thesis was primarily written by Pascal Peter in ANSI C. Cuda was used for parallelisation on the GPU. In addition, code of many other people has contributed to this work directly or indirectly. The following list contains all external code in alphabetic order.

Basic Tools: An assortment of small basic tools like memory allocation, file input/output, subsampling, Gaussian smoothing, and many more was provided by many different authors of the MIA Group.

Binary Representation of Trees: As a programming contractor, Leif Bergerhoff has written a library for efficient binary representations of subdivision trees.

Colourisation of Levin et al. For comparative experiments, we have used the reference implementation of Levin et al. [164].

Cuda Implementation of Diffusion: All of the Cuda implementations in this thesis were inspired by examples provided by Christopher Schroers.

Discretisation of Diffusion Processes: Joachim Weickert has provided a reference implementation of the discretisation scheme from [299].

DjVu: For our comparative experiments, we have used the DjVuLibre library from `http://djvu.sourceforge.net/`.

Exemplar-based Inpainting with sparse known data in Chapter 6 uses the reference implementation by Facciolo et al. [90].

Fast Explicit Diffusion: In most cases, we use the freely available implementation of the FED framework by Grewenig et al. [113].

Index Structure of the Video Codec: Christian Schmaltz contributed the implementation of the index structure that enables random access in the video codec of Chapter 7.

JBIG: For our comparative experiments, we have used the JBIG-KIT by Markus Kuhn (`https://www.cl.cam.ac.uk/~mgk25/jbigkit/`).

JBIG2: For our comparative experiments, we used the jbig2enc library maintained by Adam Langley (`https://github.com/agl/jbig2enc`).

JPEG: For our experiments with JPEG, we have used libjpeg 6.2 (`https://packages.debian.org/de/sid/libjpeg62`) via graphics magick 1.3.21 (`http://www.graphicsmagick.org/`) tool *convert*. All parameters remained at default settings.

JPEG2000: For our experiments with JPEG2000 (or more precisely, Part 1 of the standard), we have used the Jasper library v1.900.1 (`http://www.ece.uvic.ca/~frodo/jasper/`) via graphics magick 1.3.21 (`http://www.graphicsmagick.org/`) tool *convert*. All parameters remained at default settings: LRCP lossy mode with 9/7 real-to-real wavelets, 6 resolution levels, and nominal blocks of size 64×64 for Tier 1 encoding.

Optimal Control: The original reference code for the optimal control scheme for homogeneous diffusion in Chapter 5 came from Laurent Hoeltgen. Sebastian Hoffmann extended it to biharmonic diffusion.

PAQ: We have used a slightly modified version of PAQ [178] that was originally adapted by Christian Schmaltz for use in R-EED.

R-EED: The reference implementation of R-EED maintained by Christian Schmaltz was used for comparisons and as basis for the video compression codec in Chapter 7.

Stochastic Sparsification: In order to find exact masks with stochastic sparsification in Chapter 5, we have used the reference implementation by Markus Mainberger.

Stochastic Tree Densification: Frank Nedwed has provided an original implementation of stochastic tree densification and non-local node exchange that we have extended and improved for Chapter 5.

Successive Overrelaxation: In order to compare performance to FED, we have used a coarse-to-fine implementation of anisotropic diffusion by Andrés Bruhn.

MIAPDE - Towards A Unifying Framework for PDE-based Compression

In the following we provide a sketch of a unifying MIAPDE codec that could provide a common development platform for PDE-based codecs. It does not aim at a final standardisation process, yet, but would facilitate compatibility and interchangeability of standard components that are often used in PDE-based compression. Moreover, it could act as a basis for first publicly available PDE-based codecs.

The MIAPDE codec supports multiple "content types" that allow to address different kinds of image data in individual ways. For each content type, a combination of custom header information and efficiently encoded image data can be stored. The codec does not restrict the type of inpainting that is employed, i.e. it can also be used for future projects that use novel inpainting techniques or combine multiple methods. The standardisation allows to use both individual encoders/decoders for different projects as well as one encoder/decoder for all compression methods that obey the standard.

A modular block structure forms the foundation of the MIAPDE codec. A MIAPDE file always starts with with a *content ID* block that indicates what to expect from the rest of the file.

The next block is the *file header* which contains general information about the image (e.g. dimensions) and optional *parameters* for the decompression algorithm. If the decoder requires jumping addresses to either decode the rest of the file or for quick random access, the header also contains these addresses. This address list is referred to as the *index*.

Finally, the actual image data is encoded in a sequence of *data containers*. There must be at least one container (otherwise the file would be empty). Each container can optionally provide its own *container header* with information that is only relevant for decoding this container. A sequence of *data blocks* can contain different content (e.g. one block of positional data, one block of grey value data).

Content Type

The content ID is a binary number of the length one byte. It encodes two main messages: 1. what type of image content is contained in the file, 2. which decoder has to be used to decode

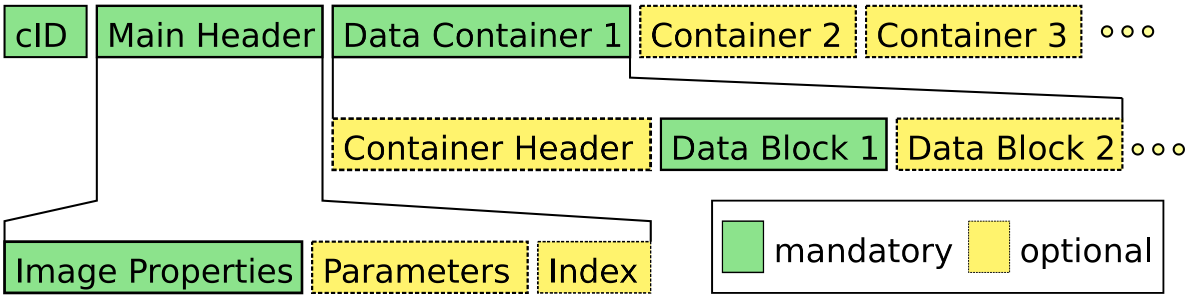


Figure I.1.: **Codec Structure**: Each PDE-file consists out of a mandatory content ID, an optional file header and at least one data container with at least one block of data.

Table I.1.: Data types supported by file headers.

Data Type	Size
Bit-Flag	1 bit
Integer	1 byte
Integer	custom
Float	custom

the image. Adding a new content type consequently implies adding a also a new corresponding decoder module to the universal decoder. Since the length is one byte, 256 different content types can be encoded, which should suffice for the lifetime of the MIAPDE-codec.

The type of image content includes the dimensionality of the data, the number of channels and, if available, information on the structure of the encoded images. Possible content types are for example: generic, cartoon, depth map, medical,...

With the associated decoder, the image can be extracted from the rest of the file. More than one decoder can be associated with the same type of image content. For example: ID 0 is associated to cartoon images with a decoder that uses homogeneous diffusion and data close to edges, encoded with JBIG. At the same time, ID 1 could also be associated to cartoon images and homogeneous diffusion, but this decoder uses scattered data instead (e.g. from an optimal control scheme as in Chapter 5).

File Header

In general, the header consists of a sequence of binary codes. This sequences is predetermined by the content ID. Each binary code belongs to a small set of predetermined data types (see Table I.1).

By design, the number of supported data types is limited in order to keep the framework simple. Arbitrary other data types can be added by individual encoders by using type conversions (e.g.

quantised floats stored as an integer) or custom bit-sequences (with the bit-flag data type). By default, image properties and parameters are stored in a raw binary format without additional entropy coding, since the overhead is often larger than the gain of compressing this data.

The index contains a list of potentially very large integer numbers. In order to store these numbers efficiently, the framework supports Golomb-coding [106].

Data Containers

The idea behind data containers is to encapsulate data that belongs together. Each container consists of a container header and a sequence of data blocks that might encode different types of data (e.g. positional data, grey value data, image gradients, ...). Each block is encoded with a suitable entropy-coder or another (potentially lossy) encoding scheme. All of the information that the decoder requires to access and decode each data block is contained in the container header. The structure of the container header is determined by the content ID and it consists out of the same data types as the file header.

Data containers are motivated by applications like progressive mode in R-EED, where a low detail and a high detail version of an image are stored in the same file. Here, one container corresponds to the low detail version and another one to the high detail version. In this example, both containers encode both positional and grey value data. These different kinds of data are entropy-coded separately and stored in individual data blocks within the corresponding container.

Outlook

Overall, the concept of the MIAPDE codec exploits the fact that many PDE-based compression codecs rely on the same or very similar building blocks. A framework that supports only a small amount of different entropy coders, inpainting operators and data types could already cover most of the current literature on PDE-based compression (see Table 3.1). In combination with standardised benchmarks for different content types, development of PDE-based codecs could be potentially accelerated. Of course, such a unifying framework needs a broad consensus about required features among researchers from the PDE-based compression community to be successful.

Colophon

This thesis was typeset with LaTeX 2$_{\varepsilon}$.
It is based on the *Clean Thesis* style developed by Ricardo Langner.
Pascal Peter performed individual stylistic and structural modifications to this basic template.

The original, unmodified *Clean Thesis* style is available at
`http://cleanthesis.der-ric.de/`.