Methods for PDE-based Image Reconstruction

A dissertation submitted towards the degree
Doctor of Natural Sciences (Dr. rer. nat.)
of the Faculty of Mathematics and Computer Science
of Saarland University

by

Vassillen Mihailov Chizhov

Saarbrücken, 2025

Date of Defence
29.09.2025

Dean of Faculty
Univ.-Prof. Dr. Roland Speicher

Chair of Committee
Prof. Dr. Peter Ochs

Reviewers
Prof. Dr. Joachim Weickert
Prof. Dr. Martin Welk

Academic Assistant
Dr. Mukul Dwivedi

Short Abstract

This thesis addresses scattered data interpolation and image approximation, focusing on reconstructions via partial differential equations (PDEs). While classical methods like JPEG and JPEG2000 are widely used, PDE-based inpainting often yields better results for images with low to medium texture content. We extend PDE-based approaches by incorporating diverse features – colours, derivatives, and local integrals – into the reconstruction process. Experiments show that these features can reduce the MSE by more than 60% without increasing the data budget. The framework also supports nonlinear operators and features. Since PDE-based methods are computationally demanding, we develop fast inpainting techniques using adaptive finite elements on the CPU, and multigrid solvers on the GPU. The latter enables real-time inpainting of 4K images. Beyond efficiency, we tackle data selection for low-error reconstructions through spatial and tonal optimisation. We propose greedy algorithms for spatial selection and fast solvers for large-scale tonal optimisation problems. We also explore the link between denoising and inpainting, showing their connection through probabilistic theory. Finally, we present perceptual optimisation techniques for Monte Carlo rendering, aiming for blue-noise error distributions that decay under low-pass filtering.

Kurzzusammenfassung

Diese Arbeit behandelt die Scattered Data Interpolation und Bildapproximation mit partiellen Differentialgleichungen (PDEs). Klassische Verfahren wie JPEG und JPEG2000 sind verbreitet, doch PDE-basiertes Inpainting liefert oft bessere Ergebnisse bei Bildern mit geringer bis mittlerer Textur. Wir erweitern PDE-Ansätze durch Einbindung von Farben, Ableitungen und lokalen Integralen. Experimente zeigen, dass diese Merkmale den mittleren quadratischen Fehler (MSE) um mehr als 60% senken können – ohne zusätzlichen Datenbedarf. Unser Ansatz unterstützt auch nichtlineare Operatoren. Da PDE-Methoden rechenintensiv sind, entwickeln wir schnelle Inpainting-Techniken mit adaptiven finiten Elementen (CPU) und Mehrgitterlösern für Grafikprozessoren (GPUs). So wird Echtzeit-Inpainting von 4K-Bildern möglich. Zur Fehlerreduktion optimieren wir die Datenauswahl räumlich und tonal. Wir schlagen Greedy-Algorithmen für die räumliche Auswahl und schnelle Löser für große tonale Optimierungsprobleme vor. Zudem etablieren wir einen Zusammenhang zwischen Entrauschen und Inpainting, der auf einer wahrscheinlichkeitstheoretischen Beziehung basiert. Abschließend präsentieren wir perzeptuelle Optimierungstechniken für Monte-Carlo-Rendering mit blauem Rauschen, das unter Tiefpassfilterung abklingt.

Abstract

This thesis is concerned with scattered data interpolation and approximation of images, with a focus on reconstructions resulting from the solution of partial differential equations (PDEs). Classical approaches for image reconstruction rely on the discrete cosine transform (DCT – JPEG) and the discrete wavelet transform (DWT – JPEG2000), and are widely used for compression. However, PDE-based inpainting has been shown to outperform the latter for images with low to medium texture content.

In our work we augment previous PDE-based approaches with the ability to combine a variety of features from which the image is to be reconstructed. Those are colours, derivatives, and local integrals. Our experiments demonstrate that those features are practically relevant as they can decrease the MSE by more than 60% for the exact same data budget as standard approaches. The framework is theoretically well-founded and it is general in the sense that it can also handle nonlinear inpainting operators and nonlinear features.

In practice, PDE-based reconstruction is computationally intensive, especially compared to approaches based on fast transforms (fast variants of the DCT and the DWT in JPEG and JPEG2000). For this reason we develop fast inpainting methods based either on reducing the problem size – by using adaptive finite element discretisations, or based on speeding up the solution process – by combining state-of-the-art solvers such as multigrid, Krylov methods, and domain decomposition, and parallelising them on the GPU. This allows us to achieve real-time inpainting of 4K images.

Except for inpainting efficiency, we are also concerned with the problem of selecting data that results in low-error reconstructions. The latter is made up of two parts: spatial optimisation and tonal optimisation. The spatial optimisation is a hard combinatorial problem for which we devise efficient greedy algorithms that result in high quality reconstructions. The tonal optimisation is a simple least squares problem, but in the context of image inpainting this problem is often very large. We design methods that leverage the sparsity of our matrices in order to achieve fast tonal optimisation. Our methods generalise to our feature inpainting framework.

Additionally, we study the relationship between denoising and inpainting, and show that the two are intricately related. An averaging of multiple inpaintings results in an approximation of a denoising process. The two are related by a rich probabilistic theory.

Last but not least, we develop algorithms for the perceptual optimisation of Monte Carlo rendering. The goal is to optimise the integration sequences such that the resulting error has a blue-noise distribution that decays quickly under low-pass filters.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Joachim Weickert, for his invaluable guidance and for giving me the opportunity to conduct engaging research within the Mathematical Image Analysis (MIA) group at Saarland University. His openness to discussing scientific challenges, his insightful suggestions for both ongoing and future research, and his generous support throughout the writing of our publications have been instrumental to my academic development.

I would also like to thank Prof. Martin Welk for kindly agreeing to serve as the external reviewer of my doctoral thesis. His detailed and constructive feedback has been invaluable in shaping this final version of the dissertation. I am also deeply appreciative of the thoughtful and engaging questions he raised during my defence, which enriched the discussion and provided valuable insights.

Next, I would like to thank my co-authors and collaborators: Dr. Iliyan Georgiev, Dr. Karol Myszkowski, and Dr. Gurprit Singh, for our joint work on perceptual error optimization for Monte Carlo rendering; Ferdinand-Dennis Jost, for our collaboration on the feature inpainting framework; Daniel Gaa, for our work on denoising by inpainting; and Niklas Kämper, for our collaboration on fast inpainting solvers and data optimisation strategies.

In addition to those direct collaborations, I would like to extend my heartfelt thanks to all other current and former colleagues and scientific staff at the MIA chair: Dr. Tobias Alt, Sarah Andris, Dr. Matthias Augustin, Dr. Marcelo Cárdenas, Michael Ertel, Peter Franke, Hyoseung Kang, Andrea Kreutzer, Dr. Rahul Mohideen Kaja Mohideen, Dr. Pascal Peter, Kristina Schaefer, Jan Schmitz, Karl Schrader, Jón Arnar Tómasson and Ellen Wintringer. Your presence and contributions have made the research environment both stimulating and enjoyable.

Above all, my deepest gratitude goes to my parents, Prof. Mihail Vladimirovich Chizhov and Prof. Daniela Petrova Kirilova, whose unwavering support and encouragement made it possible for me to pursue my doctoral studies. I am also sincerely thankful to my siblings, Emanuil Mihailov Chizhov and Dr. Rosanna Mihailova Chizhova-Sturgeon, for always bringing light and joy into my days.

Last but not least, I gratefully acknowledge that a large part of this work has been financed by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 741215, ERC Advanced Grant INCOVID).

Content

Chapter 1

Introduction

This thesis addresses the fundamental problem of function approximation, which underlies a variety of tasks in image processing and computer graphics. The primary objective of this work is to develop practical and efficient algorithms for sparsely approximating image data. Before presenting these algorithms in the following chapters, we begin by introducing the theoretical setup of the approximation problem.

1.1 Data Optimisation

Let the function we wish to approximate be $f \in \mathcal{V}$, from some space of functions $\mathcal{V}$ (typically L^2 or $\mathbb{R}^N$) with domain Ω and codomain $\mathbb{R}^p$ (or $\mathbb{C}^p$). The function f may represent a continuous image $f : [0,1]^2 \to \mathbb{R}$, or a discrete digital image $f : \{0,\ldots,W-1\} \times \{0,\ldots,H-1\} \to \mathbb{R}$. It could also represent the parametrisation of a surface $\boldsymbol{f} : \Omega \to \mathbb{R}^3$, or the locations of the vertices $\boldsymbol{f} : \{0,\ldots,N-1\} \to \mathbb{R}^3$ of a triangular piecewise-linear mesh surface. In any case, our goal will be to select a finite-dimensional, $\dim \mathcal{U}_{\mathcal{X}} = m$, (potentially nonlinear) subspace $\mathcal{U}_{\mathcal{X}} \subseteq \mathcal{V}$, parametrised by some set $\mathcal{X}$ that minimises the difference between f and $v \in \mathcal{U}_{\mathcal{X}}$ w.r.t. some given metric $\epsilon : \mathcal{V} \times \mathcal{V} \to [0,\infty)$

$$\min_{|\mathcal{X}|=m} \min_{v \in \mathcal{U}_{\mathcal{X}}} \epsilon(v,f). \tag{1.1}$$

For the moment we take $|\mathcal{X}| = m$ to be shorthand for: $\mathcal{X}$ such that $\dim \mathcal{U}_{\mathcal{X}} = m$. When we introduce interpolating reconstruction, in Section 1.2, $\mathcal{X}$ will be a set of finitely many points and $|\mathcal{X}| = m$ will be the number of those points. It is a data budget constraint, e.g., m can be the number of pixels we want to store from an image, or the number of vertices we want to store from a triangular mesh. We term the minimisation over $\mathcal{X}$ – *spatial optimisation*, and the inner minimisation – *tonal optimisation*. We elaborate on the motivation for this terminology in Section 1.3 and Section 1.4.

While there are many reasonable error metrics, throughout most of this thesis we consider the 2-norm since it is induced by an inner product and results in a simple problem structure. Moreover the mean-squared-error (MSE) is widely used in image processing and computer graphics. In Chapter 6 we also consider more intricate perceptual metrics such as HDR-VDP-2 [161]. The choice of space $\mathcal{U}_{\mathcal{X}}$ is a non-trivial task and has a great effect on the approximation quality and the ease of finding an optimum. The inner minimisation problem (which we term *tonal optimisation*)

$$u \in \operatorname*{arg\,min}_{v \in \mathcal{U}_{\mathcal{X}}} \|v - f\|_2^2, \tag{1.2}$$

is typically much easier to solve. If $\mathcal{U}_{\mathcal{X}}$ is a linear space then it always results in a linear system that has at least one solution. If the space $\mathcal{U}_{\mathcal{X}}$ is nonlinear we get a nonlinear system of equations. In order to make the above concrete we next discuss the spaces $\mathcal{U}_{\mathcal{X}}$ that we consider throughout the thesis.

1.2 Interpolating Reconstruction

Given a set of points $\mathcal{X} = \{\boldsymbol{x}_i \in \Omega \subseteq \mathbb{R}^d : 1 \leq i \leq m\}$, assume that there is a unique function $u \in \mathcal{U}_{\mathcal{X}}$, such that it interpolates f at the data points: $u(\boldsymbol{x}_i) = f(\boldsymbol{x}_i)$, $1 \leq i \leq m$. Throughout this work, we will often use the term *inpainting* interchangeably with reconstruction, particularly in the context of image reconstruction. Specifically, we refer to the reconstruction of an image from a sparse set of points as *sparse image inpainting* (see for example [89]). This stands in contrast to classical non-sparse image inpainting [30], which typically presents a less challenging and more well-posed problem.

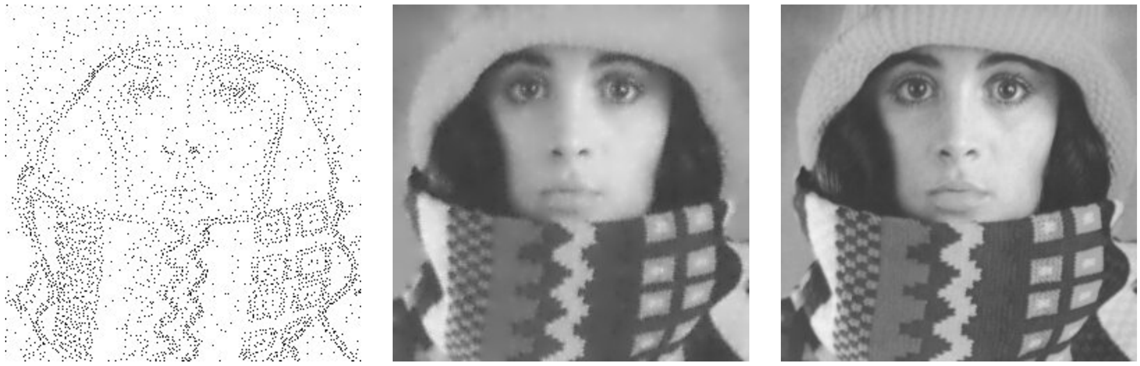
1.2.1 Polyharmonic Inpainting

As a concrete example of a space $\mathcal{U}_{\mathcal{X}}$ consider the space of q-harmonic functions on $\Omega \setminus \mathcal{X}$ with reflecting boundary conditions on $\partial\Omega$ which interpolate arbitrary coefficients on $\mathcal{X}$:

$$\begin{aligned} (-\Delta)^q u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Omega \setminus \mathcal{X}, \\ \partial_{\boldsymbol{n}}(-\Delta)^\ell u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega,\ 0 \leq \ell \leq q-1, \\ u(\boldsymbol{x}) &= f(\boldsymbol{x}), && \boldsymbol{x} \in \mathcal{X}, \end{aligned} \tag{1.3}$$

where $\boldsymbol{n}$ is the normal to the boundary $\partial\Omega$. The usage and study of polyharmonic splines for scattered data interpolation dates back to Harder and Desmarais [100], Duchon [76], and Meinguet [165], albeit they consider an unbounded domain. A more recent work applying [165] to inpainting is the paper by Kalmoun and Nasser [125]. For more details on polyharmonic boundary value problems see the book by Gazzola, Grunau, and Sweers [91].

For us the key point is that by the Sobolev embedding theorem we have that $H^q(\Omega) \subset C^r(\overline{\Omega})$, if $q > r + \frac{d}{2}$. That is, $H^q(\Omega)$ is a reproducing kernel Hilbert space (RKHS) if q is greater than half of the dimension of Ω. This implies, for instance, that harmonic inpainting with pointwise interpolation is well-posed in 1D, but not in 2D, since $H^1(\Omega)$ is not an RKHS in 2D. Indeed the Green's functions of the Laplacian have logarithmic singularities for $d \geq 2$ – this applies also to our setting where Ω is a rectangle [166]. If we discretise the harmonic inpainting problem we recover well-posedness in 2D as long as



mask $\boldsymbol{c}$ harmonic inpainting $\boldsymbol{u}$ original image *trui* $\boldsymbol{f}$

FIGURE 1.1: Harmonic inpainting with a 5% mask. Inpainting MSE: 38.24.

$\mathcal{X} \neq \varnothing$. However, the discrete reconstruction does inherit artifacts from the logarithmic singularities in the continuous setting. The discrete counterpart of (1.3) is given as:

$$\begin{aligned} (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L}^q \boldsymbol{u} &= \boldsymbol{0}, \\ \boldsymbol{C}\boldsymbol{u} &= \boldsymbol{C}\boldsymbol{f}, \end{aligned} \tag{1.4}$$

where $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$ is the *mask matrix* constructed from the *(inpainting) mask* $\boldsymbol{c} \approx \boldsymbol{1}_{\mathcal{X}}$, and $\boldsymbol{L} \approx -\Delta$ approximates the negated Laplacian (typically with the 5-point stencil) with reflecting boundary conditions. Often for convenience we combine the two equations:

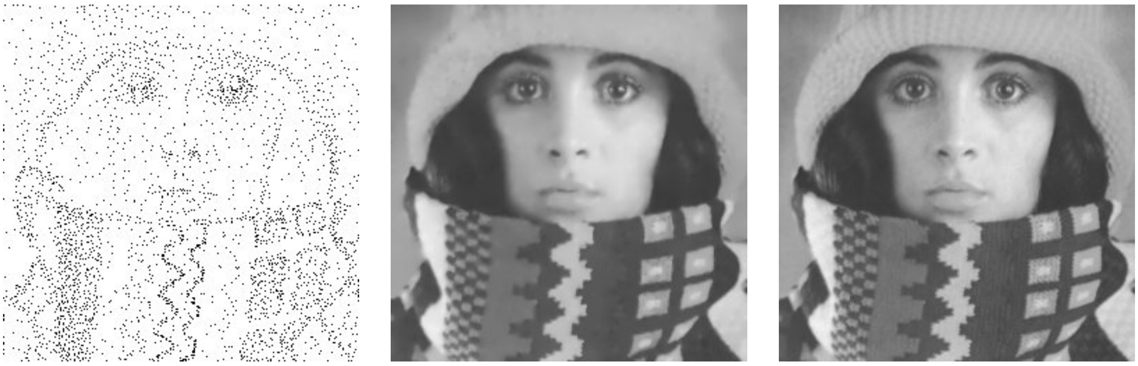
$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L}^q)\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}. \tag{1.5}$$

However, we will see that this is not always feasible when we generalise the interpolation data. The above problem has a unique solution when $\boldsymbol{c} \neq \boldsymbol{0}$, since the kernel of $\boldsymbol{L}$ is spanned by $\boldsymbol{1}$ and a single interpolation point is sufficient to fix the mean as long as Ω is connected. If Ω consists of several connected components, and $\boldsymbol{L}$ discretises the negated Laplacian with reflecting boundary conditions on those, we need at least one interpolation point per component. But then we can just decompose the problem into as many decoupled problems as there are components. An illustration of harmonic inpainting is provided in Figure 1.1. We also note that polyharmonic inpainting of surfaces makes the matrix $\boldsymbol{L}$ dependent on $\boldsymbol{u}$ since the Laplace-Beltrami operator depends on the surface's geometry. That is of course the case if we use the Laplace-Beltrami operator – if we take the topological Laplacian instead, the problem remains linear.

1.2.2 Edge-Enhancing Diffusion Inpainting

As a representative of a nonlinear space $\mathcal{U}_{\mathcal{X}}$ we consider the inpaintings derived from edge-enhancing diffusion (EED) [236]:

$$\begin{aligned} -\operatorname{div}(\boldsymbol{D}(\nabla u_\sigma)\nabla u) &= 0, && \text{on } \Omega \setminus \mathcal{X}, \\ \partial_{\boldsymbol{n}} u &= 0, && \text{on } \partial\Omega, \\ u &= f, && \text{on } \mathcal{X}. \end{aligned} \tag{1.6}$$



mask $\boldsymbol{c}$ EED inpainting $\boldsymbol{u}$ original image *trui* $\boldsymbol{f}$

FIGURE 1.2: Illustration of EED inpainting with a 5% mask. Inpainting MSE: 14.95.

Here u_σ denotes a Gaussian-smoothed version of u, defined as $u_\sigma = K_\sigma * u$, where K_σ is a Gaussian kernel with standard deviation σ, and the convolution is performed with reflecting boundary conditions over the domain Ω. The vector $\boldsymbol{n}$ is the normal on $\partial\Omega$, and we have used a homogeneous Neumann (reflecting) boundary condition[1]. The diffusion tensor $\boldsymbol{D}$ is constructed from the structure tensor $\boldsymbol{J} = \nabla u_\sigma \nabla u_\sigma^\top$, by setting all of its eigenvalues to 1, except for the largest one. A diffusivity function is applied to the largest eigenvalue $\mu_{\max}$ – we typically use the Charbonnier diffusivity with a contrast parameter $\lambda > 0$:

$$\mu'_{\max} = \frac{1}{\sqrt{1 + \mu_{\max}/\lambda^2}}. \tag{1.7}$$

Incidentally, if $\mu_{\max} = |\nabla u|^2$, the above corresponds to a term that appears in the Laplace-Beltrami operator restricted to heightfields/function graphs. We use the discretisation of anisotropic diffusion from [241], which results in the quasi-linear system of equations:

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L}(\boldsymbol{u}))\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}. \tag{1.8}$$

The matrix $\boldsymbol{L}(\boldsymbol{u})$ is now a function of $\boldsymbol{u}$. That is, we have a quasi-linear system of equations. An illustration of the inpainting is shown in Fig. 1.2. Note that the MSE is less than half of that for harmonic inpainting despite of the fact that we store the same amount of data. This illustrates the quality improvements that one can achieve by considering suitable nonlinear spaces $\mathcal{U}_{\mathcal{X}}$. A more generic nonlinear inpainting operator $\boldsymbol{\psi} : \mathbb{R}^N \to \mathbb{R}^N$ (not necessarily quasi-linear) results in the following formulation:

$$\boldsymbol{C}\boldsymbol{u} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{C}\boldsymbol{f}. \tag{1.9}$$

1.2.3 Feature Inpainting

In our work we generalise the interpolation problem to not only interpolate pointwise values, but also other features such as: weighted local integrals $\int_\Omega wu \,\mathrm{dvol} = \int_\Omega wf \,\mathrm{dvol}$,

[1] In the above we have used $\partial_{\boldsymbol{n}} u(\partial\Omega) = 0$ since this is what previous inpainting implementations use. It should be noted, however, that $\partial_{\boldsymbol{Dn}} u(\partial\Omega) = 0$ is a more natural boundary condition, that actually reflects the flux across the boundary.

derivatives $\partial_x u = \partial_x f$, or even nonlinear features. In the discrete setting let $\boldsymbol{\phi}_i : \mathbb{R}^N \to \mathbb{R}^N$ for $1 \le i \le k$ be a family of (potentially nonlinear) functions, and let $\boldsymbol{c}_i$ be their corresponding masks. We can collect those into vectors of k times the image size N:

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi}_1^\top & \cdots & \boldsymbol{\phi}_k^\top \end{bmatrix}^\top : \mathbb{R}^N \to \mathbb{R}^{k \cdot N}, \quad \boldsymbol{c} = \begin{bmatrix} \boldsymbol{c}_1^\top & \cdots & \boldsymbol{c}_k^\top \end{bmatrix}^\top \in \mathbb{R}^{k \cdot N}, \quad \boldsymbol{C} = \operatorname{diag}(\boldsymbol{c}). \tag{1.10}$$

Then we can write the feature inpainting problem as follows:

$$\big(\boldsymbol{I} - \boldsymbol{P}(\boldsymbol{u})\big)\boldsymbol{\psi}(\boldsymbol{u}) = 0, \tag{1.11}$$

$$\boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{u}) = \boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{f}), \tag{1.12}$$

where $\boldsymbol{I} - \boldsymbol{P}(\boldsymbol{u}) = \boldsymbol{I} - (\boldsymbol{C}\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}))^{+}\boldsymbol{C}\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u})$ is the orthogonal projection matrix onto the kernel of the Jacobian of the constraints (we denote the Moore-Penrose inverse of $\boldsymbol{A}$ with $\boldsymbol{A}^{+}$). We will not go into the details of the derivation of the above formulation here, and instead we defer this to Chapter 2. Currently it suffices to know that in the most general setting that we consider, the subspace $\mathcal{U}$ in which we approximate is technically a function of $\boldsymbol{c}, \boldsymbol{\psi}, \boldsymbol{\phi}$: $\mathcal{U}_{\boldsymbol{c},\boldsymbol{\psi},\boldsymbol{\phi}}$. In the continuous setting it is a function of $\mathcal{X}, \psi, \phi$: $\mathcal{U}_{\mathcal{X},\psi,\phi}$. In the simplest case of linear inpainting with pointwise value interpolation we have $\psi(u) = Lu$ and $\phi(u) = u$.

1.3 Tonal Optimisation

Consider the original data optimisation problem with the 2-norm as an error metric

$$\min_{|\mathcal{X}|=m} \min_{v \in \mathcal{U}_\mathcal{X}} \|v - f\|_2^2, \tag{1.13}$$

where $\mathcal{U}_\mathcal{X} \equiv \mathcal{U}_{\mathcal{X},\psi,\phi}$ for some fixed inpainting operator ψ and feature families ϕ. If ψ and ϕ are linear, then $\mathcal{U}_\mathcal{X}$ is a linear space. Then there exists a (potentially overcomplete) basis

$$\boldsymbol{B} = \begin{bmatrix} b_1 & \cdots & b_m \end{bmatrix} \in \mathcal{V}, \quad \mathcal{U}_\mathcal{X} = \left\{ v(\boldsymbol{x}) = \sum_{j=1}^{m} g^j b_j(\boldsymbol{x}) \ : \ \boldsymbol{g} \in \mathbb{C}^m \right\}. \tag{1.14}$$

Plugging this into the inner minimisation we get the linear least-squares problem:

$$\min_{v \in \mathcal{U}_\mathcal{X}} \|v - f\|_2^2 = \min_{\boldsymbol{g} \in \mathbb{C}^m} \left\| \sum_{j=1}^{m} g^j b_j - f \right\|_2^2 = \sum_{i=1}^{m} \sum_{j=1}^{m} \langle b_i, b_j \rangle \overline{g^i} g^j - 2 \sum_{i=1}^{m} \overline{g^i} \langle b_i, f \rangle + \|f\|_2^2, \tag{1.15}$$

where $\|v\|_2^2 = \langle v, v \rangle$. Differentiating and setting to zero yields the normal equations:

$$\boldsymbol{G}\boldsymbol{g} = \boldsymbol{s}, \quad G_{ij} = \langle b_i, b_j \rangle, \quad s_i = \langle b_i, f \rangle. \tag{1.16}$$

The latter always have a solution, but the solution is unique only when the Gramian $\boldsymbol{G}$ is invertible (i.e., the chosen interpolation points and features are not redundant). In the nonlinear setting, let us denote the reconstruction as $u(\mathcal{X}, \boldsymbol{g})$, where $\phi(u)|_\mathcal{X} = \boldsymbol{g}$, and $\psi(u) = 0$ holds on the kernel of the differential of the constraints. Then we have the nonlinear space

$$\mathcal{U}_{\mathcal{X},\psi,\phi} = \{u(\mathcal{X},\boldsymbol{g}) : \Omega \to \mathbb{C} : \boldsymbol{g} \in \mathbb{C}^m\}. \tag{1.17}$$

If the reconstruction $u(\mathcal{X},\boldsymbol{g})$ is differentiable in the argument for the $\boldsymbol{g}$ variable, then the columns of the Jacobian w.r.t. $\boldsymbol{g}$ provide a (potentially overcomplete) basis for the tangent space. Then the counterpart to the linear normal equations are the nonlinear normal equations:

$$\langle(\partial_{\mathbf{e}_i} u)(\mathcal{X},\boldsymbol{g}), u(\mathcal{X},\boldsymbol{g}) - f\rangle = 0, \quad 1 \le i \le m. \tag{1.18}$$

That is, we want the error to be orthogonal to the tangent space – the linear setting is a special case of this, where the tangent space can be assigned a constant basis. Note that in the nonlinear setting the system is nonlinear which makes the problem much more challenging.

In image processing and compression literature the inner minimisation problem

$$\boldsymbol{g}^*(\mathcal{X}) \in \operatorname*{arg\,min}_{\boldsymbol{g}\in\mathbb{C}^m} \|u(\mathcal{X},\boldsymbol{g}) - f\|_2^2, \tag{1.19}$$

is known as the *tonal optimisation* problem. This is because for an inpainting with $\phi = \mathrm{id}$, we are interpolating greyscale or colour values, i.e., tonal data. So the optimisation of the coefficients $\boldsymbol{g}$ corresponds to the optimisation of tonal data. Of course, with our extension of inpainting to other features, the term "tonal optimisation" becomes somewhat of a misnomer, but we nevertheless use it throughout the thesis.

1.4 Spatial Optimisation

Consider the original data optimisation problem:

$$\min_{|\mathcal{X}|=m} \min_{v\in\mathcal{U}_{\mathcal{X}}} \|v - f\|_2^2. \tag{1.20}$$

This time we focus on the outer minimisation. To that end we suppose that we have a procedure that allows us to compute a minimiser for the tonal optimisation problem

$$\boldsymbol{g}^*(\mathcal{X}) \in \operatorname*{arg\,min}_{\boldsymbol{g}\in\mathbb{C}^m} \|u(\mathcal{X},\boldsymbol{g}) - f\|_2^2. \tag{1.21}$$

We can then rewrite (1.20) as an optimisation only over $\mathcal{X}$

$$\min_{|\mathcal{X}|=m} \|u(\mathcal{X},\boldsymbol{g}^*(\mathcal{X})) - f\|_2^2. \tag{1.22}$$

The optimisation over $\mathcal{X}$ is typically termed *spatial optimisation* as we are optimising over the locations of the points in $\mathcal{X}$. If we were to reduce the choice of the m points in $\mathcal{X}$ to be chosen from a fixed set of N possible locations $\mathcal{X}_N$ (e.g. the pixel grid of an image, or the vertices of a triangular mesh), this becomes a combinatorial problem with search space of size

$$\binom{N}{m} = \frac{N!}{m!\,(N-m)!}. \tag{1.23}$$

The exhaustive search is intractable already in the setting of small images. For example, with $N = 256 \times 256$ and a reasonably low interpolation point density, $m = \lfloor 5\% \cdot N \rfloor$, we get an incredibly large search space of size

$$\binom{N}{m} = \binom{256 \times 256}{\lfloor 256 \times 256 \times 20^{-1} \rfloor} = \binom{65536}{3276}. \tag{1.24}$$

Since the search space grows very quickly with the resolution, in practice we resort to various (meta-)heuristics and greedy algorithms in order to find a solution close to the minimiser in reasonable time.

If $\mathcal{U}_{\mathcal{X}}$ is an interpolating space over $\mathcal{X}$, for efficiency purposes we can choose to substitute the tonally optimal coefficients $\boldsymbol{g}^*(\mathcal{X})$ with interpolating ones $\phi(f)|_{\mathcal{X}}$:

$$\mathcal{X}^* \in \operatorname*{arg\,min}_{|\mathcal{X}|=m} \|u(\mathcal{X}, \phi(f)|_{\mathcal{X}}) - f\|. \tag{1.25}$$

Most often it is (1.25) that is referred to as *spatial optimisation.* One can then apply a tonal optimisation step given the fixed $\mathcal{X}^*$ from the above problem. This corresponds to swapping the order of the minimisation problems – first spatial then tonal:

$$\boldsymbol{g}^* \in \operatorname*{arg\,min}_{\boldsymbol{g} \in \mathbb{C}^m} \|u(\mathcal{X}^*, \boldsymbol{g}) - f\| \quad \text{such that} \quad \mathcal{X}^* \in \operatorname*{arg\,min}_{|\mathcal{X}|=m} \|u(\mathcal{X}, f|_{\mathcal{X}}) - f\|, \tag{1.26}$$

compared to the original data optimisation problem (1.20) – first tonal then spatial:

$$\mathcal{X}^* \in \operatorname*{arg\,min}_{|\mathcal{X}|=m} \|u(\mathcal{X}, \boldsymbol{g}^*(\mathcal{X})) - f\| \quad \text{such that} \quad \boldsymbol{g}^*(\mathcal{X}) \in \operatorname*{arg\,min}_{\boldsymbol{g} \in \mathbb{C}^m} \|u(\mathcal{X}, \boldsymbol{g}) - f\|. \tag{1.27}$$

Theoretically the global minimiser of (1.26) is worse as a minimiser of (1.20), since it is minimising an objective with the minimisation order flipped. However, in practice the formulation in (1.26) is much more efficient to work with when using greedy algorithms, and counter-intuitively those often produce better results (see [111]) when applied to (1.26). This is an instance where **optimising an inexact model is much easier, and thus yields a better approximate solution than optimising the exact model, which may lead to a poorer approximate result – despite the exact model theoretically having a superior (or at least not inferior) global minimum**. For this reason we discuss only interpolating spatial optimisation in Chapter 4.

The relationship of (1.26) to (1.27) is similar to the relationship between matching pursuit and orthogonal matching pursuit [158, 200]. Speaking of the latter, in Section 1.5 we briefly discuss when our data optimisation problem (1.20) can be brought to the optimisation formulation appearing in the literature on pursuit algorithms.

1.5 Data Optimisation Formulation in Pursuit Algorithms

In practice, the problem in (1.20) can be related to the sparse linear synthesis-based approximation problem:

$$\min_{\|\boldsymbol{g}\|_0 = m} \|u - f\|_2^2, \quad u(\boldsymbol{x}) = \sum_{j=1}^{N} g^j b_j(\boldsymbol{x}), \tag{1.28}$$

for which the classical pursuit algorithms [158, 200, 210] (matching pursuit, orthogonal matching pursuit, basis pursuit, etc.) were developed[2]. We can relate the above formulation to our interpolating analysis-based formulations if we have $b_i(\boldsymbol{x}_j) = \delta_{ij}$ for $1 \le i, j \le N$, where $\boldsymbol{x}_j \in \mathcal{X}_N$. Then $\mathcal{U}_{\mathcal{X}} \subseteq \mathcal{U}_{\mathcal{X}_N}$ for $\mathcal{X} \subseteq \mathcal{X}_N$. As a special case this is true for discretisations of the polyharmonic problem (1.4). In the discrete setting **any (well-posed) analysis-based interpolating linear inpainting is equivalent to a transform-based reconstruction**:

$$\boldsymbol{Cu} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{Lu} = \boldsymbol{Cf} \iff \boldsymbol{u} = \boldsymbol{L}^{+}\boldsymbol{Ca} + \boldsymbol{V}_{\text{ker}}\boldsymbol{\mu},\ \boldsymbol{U}_{\text{ker}}^{*}\boldsymbol{Ca} = \boldsymbol{0},\ \boldsymbol{Cu} = \boldsymbol{Cf},\ \boldsymbol{L} = \boldsymbol{U\Sigma V}^{*}, \tag{1.29}$$

where $\boldsymbol{L}$ plays the role of the transform, and the Moore-Penrose inverse $\boldsymbol{L}^{+}$ plays the role of the inverse transform, which is used for the reconstruction up to vectors in the kernel of $\boldsymbol{L}$. If $\boldsymbol{L}$ is a discretisation of a linear differential operator (along with the boundary conditions) then $\boldsymbol{L}^{+}$ is a discrete counterpart of its Green's functions – **the inpainting-based and transform-based formulations are spectral counterparts of each other**. This is not the case any more when one considers more general reconstructions such as through linear splines on a Delaunay triangulation [70] (this is a synthesis instead of an analysis formulation). There $\mathcal{X} \subseteq \mathcal{X}_N$ implies $\mathcal{U}_{\mathcal{X}} \subseteq \mathcal{U}_{\mathcal{X}_N}$ only if the triangulations are hierarchical – which is generally untrue for Delaunay triangulations. The correspondence also fails in a nonlinear analysis formulation such as in edge-enhancing diffusion inpainting (Green's functions and the SVD are a linear concept), or if we consider generalised interpolation of multiple families of features.

1.6 Own Contributions

Our main contributions are published in the proceedings [58, 120] from two conferences, and in three journal papers [59, 88, 128]. I have presented results from the above works at CAIP21 [58], SIGGRAPH22 [59], GAMM23 (unpublished extensions of [58]), and ICASSP23 [120]. Below I list a brief summary of the publications on which this work is based, as well as the corresponding chapters in the thesis.

Optimising Different Feature Types for Inpainting-Based Image Representations. In [120], we present the first general framework for inpainting with arbitrary

[2] Often in the literature the budget constraint and error minimisation appear flipped, but we can also do so in our formulations if the goal is not a fixed budget but rather achieving a specific error threshold.

features that can be expressed as linear equality constraints. This framework is built upon a reformulation of harmonic inpainting as a constrained optimisation problem, enabling us to combine the traditional pointwise colour interpolation constraints with more general linear equality constraints. The solution is obtained via the Karush–Kuhn–Tucker (KKT) conditions, resulting in a symmetric but indefinite linear system, which we solve using the SYMMLQ algorithm [180]. Details and extensions of the inpainting framework to the setting of nonlinear operators and nonlinear constraints are described in Chapter 2 of the thesis.

More significantly, we develop efficient spatial and tonal optimisation strategies tailored to this novel inpainting formulation. The spatial optimisation automatically allocates the data budget across different feature types. We identify a meaningful set of features – colour values, derivatives, and local integrals – that enable us to substantially improve the quality of the reconstruction. A description of the data optimisation and extensions of it are presented in Chapter 4.

My main contributions to [120] include implementing and testing various solvers for the indefinite system, theoretical insights into the reformulation of the problem as an optimisation problem, the design of the spatial and tonal optimisation algorithms, and a substantial portion of the programming code. The original idea was proposed by Prof. Joachim Weickert, and the initial theoretical groundwork and code were primarily developed by the first author, Ferdinand Jost. The final version of the code used to generate the results was mostly based off my work. The contents in Chapter 2 and Chapter 4 represent a substantial extension of the work presented in [120], and constitute original research that is my own contribution. Two figures (Fig. 4.2 and Fig. 4.4) are reproduced from the original work [120], while all other experiments on feature inpainting are new and use the improvements and extensions discussed in the thesis.

Efficient Data Optimisation for Harmonic Inpainting with Finite Elements. In [58] we extend sparse image inpainting with homogeneous diffusion by using finite elements (FE). This considerably decreases the runtime compared to prior approaches since we are able to greatly reduce the number of unknowns, while preserving a high quality solution owing to the adaptivity of our FE mesh.

We also reformulate the tonal optimisation problem such that a nested conjugate gradient solver is applicable, where the outer iterations optimise the tonal values, and the inner iterations solve inpainting-like problems. For the spatial optimisation problem we combine ideas from a previous approach based on error map dithering [131] and Voronoi densification [68], and devise a Delaunay densification approach that allows a trade-off between runtime and quality. Taken together, these allow us to efficiently perform inpainting and optimise the reconstruction data on the CPU, with computational efficiency that is orders of magnitude better than prior approaches. Chapter 3 and Chapter 4 include parts of this work, and the extensions presented at GAMM23 are briefly discussed in Section 3.1.7.

The main theoretical developments and all experiments presented in this paper are my own. The initial idea, as well as valuable discussions and guidance in refining the manuscript, were provided by Prof. Joachim Weickert.

Efficient Parallel Algorithms for Inpainting-Based Image Representations of 4K Images. In [127] (note that this work currently remains unpublished, however, parts of it are used in our published work on data optimisation [128]), we target state-of-the-art efficiency for homogeneous diffusion inpainting by exploiting GPU parallelism and judiciously integrating concepts from advanced solvers, including domain decomposition methods, multigrid techniques, matrix-free approaches, and Krylov subspace methods. The resulting performance surpasses all previous methods by several orders of magnitude, enabling real-time inpainting of 4K images at over 60 frames per second using homogeneous diffusion. The part from [127] to which I contributed the most is reproduced in Chapter 3, along with the related work, experiments, and conclusion from the original paper.

The work on domain decomposition stems from the first author, Niklas Kämper, and thus I only briefly mention it in Chapter 3. My primary contribution involved the design and implementation of the downsampling operator for the multigrid framework that reduces leakage at coarser levels by incorporating the neighbourhood of masked pixels, as well as its theoretical justification. Additionally, I contributed through discussions with the first author and by assisting in the preparation of the manuscript. The initial idea of integrating domain decomposition methods for inpainting was suggested by Prof. Joachim Weickert.

Efficient Parallel Data Optimisation for Homogeneous Diffusion Inpainting of 4K Images. In [128], we propose data optimisation techniques for homogeneous diffusion inpainting that significantly outperform previous methods in both speed and quality. Our approach integrates the efficient inpainting algorithm from [127] into both the tonal and spatial optimisation. The spatial optimisation builds upon our method introduced in [58], but incorporates a fast GPU-based Delaunay triangulation using the jump flood algorithm [204]. Additionally, we employ an improved initialisation strategy based on dithering the Laplacian magnitude [28], which enhances quality, particularly when using a limited number of densification iterations.

The tonal optimisation extends the nested conjugate gradient framework from [58], significantly accelerating it through GPU-based domain decomposition. A high-quality initial guess is generated by interpolating local averages over the Voronoi partition, in a manner similar to the approach in [120]. These improvements enable both spatial and tonal optimisation on 4K images in under half a second.

I suggested and further extended the idea of exploiting the sparsity in the tonal optimisation based on my previous work [58] and improvements thereof. The domain decomposition parts entirely belong to the first author Niklas Kämper. Other contributions of mine include the theoretical justification for the tonal optimisation's initial guess, its connection to local average interpolation, and the interpretation of the algorithm as a Richardson iteration. I also contributed CPU implementations, stopping criteria, visual results, and engaged in extensive discussions on both the tonal optimisation and Delaunay-based densification. Generalisations of the tonal and spatial optimisation methods from [128], to the setting of the feature inpainting framework, are presented in Chapter 4. These generalisations represent original scientific contributions that I developed as part of this thesis. I do not discuss domain decomposition aspects since the work on those belongs to the first author Niklas Kämper.

Connecting Image Inpainting with Denoising in the Homogeneous Diffusion Setting. In [88] we study the relationship between inpainting and denoising. While at first glance unrelated, it turns out that probabilistic estimators based on inpainting operators can be interpreted as denoising methods. We provide a rich probabilistic theory which we support with a multitude of experiments. Chapter 5 is a reproduction of our work [88] with very minor changes.

Most of the formalisation of the probabilistic theory (Section 5.3.1) was contributed by me, along with some experiments and code. However, the final experimental results presented in the paper stem from code and experiments carried out by the first author – Daniel Gaa. The acceleration by low-discrepancy sequences and the proofs in the appendix are also my contribution. All other results were contributed by my other co-authors.

Perceptual Error Optimisation for Monte Carlo Rendering. In [59] we improve the perceptual quality of Monte Carlo rendering. We achieve this by optimising the seeds of the integration sequences used in the (quasi-)Monte Carlo numerical estimation of the light transport integrals in the formal solution of the rendering equation. The seeds are optimised w.r.t. a simple perceptual metric, to which we apply a set of algorithms that provide varying trade-offs between quality and speed, showing substantial improvements over prior state-of-the-art. The algorithms are able to achieve a Monte Carlo noise distribution that quickly decays under convolution with low-pass filters. We conduct evaluations using both quantitative and error-based metrics. Chapter 6 is a reproduction of our publication [59] with minor modifications.

All experiments were implemented and conducted by me. I also developed the main theoretical concepts and their formulations. I gratefully acknowledge the valuable discussions and substantial assistance in writing the manuscript provided by my co-authors: Dr. Iliyan Georgiev, Dr. Karol Myszkowski, and Dr. Gurprit Singh. The initial idea for the work was suggested by Dr. Gurprit Singh and was based off prior published work by Georgiev and Fajardo [93], and Belcour and Heitz [103].

1.7 Organisation of the Thesis

In Chapter 1, we introduce the data optimisation problem, present the main inpainting operators used throughout the thesis, list our contributions, and summarise the thesis structure.

In Chapter 2, we introduce a generalisation of the inpainting framework that accommodates linear and nonlinear inpainting operators and linear and nonlinear features, which can be formulated in terms of equality constraints. We present experimental results demonstrating that **the additional features yield significant quality improvements (in some cases more than 60% MSE reduction)**. While this chapter is partially based on our work [120] presented at ICASSP23, it constitutes a substantial generalisation of the results presented in the conference paper. The extended formulation does not require that the inpainting operator is linear or that it is the gradient of an energy functional – this is crucial as EED inpainting is neither [244]. Furthermore, we extend the framework to handle nonlinear features, and develop a sequential quadratic programming (SQP)-inspired trust region approach in order to solve the resulting inpainting problem.

In Chapter 3, we discuss our fast inpainting approaches based on: the finite element method (FEM) [58], and our multigrid strategy for the finite difference method (FDM) [127, 128]. A brief introduction to FEM is provided in Section 3.1. This is followed by experimental results demonstrating the quality improvements achieved using the FEM framework. Notably, **efficiency gains of over 10 times can be achieved on the CPU even for small images**. Additional results related to the efficiency improvements and resolution scaling in the context of data optimisation are deferred to Chapter 4. Section 3.1.7 outlines several extensions that significantly broaden the scope of the methods introduced in [58]. In Section 3.2, we present our multigrid-based fast inpainting approach from [127, 128], which, when combined with domain decomposition, **enables harmonic inpainting of 4K images at over 60 frames per second on the GPU**. With only minor modifications, Section 3.2 reproduces the part on multigrid from our journal publication [128], as well as the related work, experiments, and conclusion.

Our algorithms for tonal and spatial optimisation are presented in Chapter 4. They subsume the approaches from our FEM work [58], our efficient data optimisation methods [128], and the data optimisation framework for feature inpainting from [120]. The algorithms described in the chapter are **applicable to our feature inpainting framework with its extensions from Chapter 2, they achieve a very good runtime-to-quality ratio, and scale linearly in terms of the image resolution.** The chapter is a natural conclusion to the discussion of generalised inpainting from Chapter 2 and the fast inpainting methods from Chapter 3.

Chapter 5 reproduces our journal publication on denoising by inpainting [88], with only small changes implemented for consistency. It explores the theoretical connection between the seemingly distinct processes of denoising and inpainting. The theory offers both deterministic and probabilistic interpretations, showing how an average of multiple inpaintings can act as a denoising process. **The key result is a relationship between the mask density in homogeneous diffusion inpainting and the diffusion time in homogeneous diffusion denoising.** Within the context of this thesis, this chapter serves as a conceptual bridge between the inpainting algorithms discussed in the preceding chapters and their extension to denoising processes.

Chapter 6 reproduces our published work on perceptual error optimisation for Monte Carlo rendering [59], with minor formatting and consistency-related adjustments. This work focuses on optimising the integration sequences in Monte Carlo rendering to produce error distributions that decay rapidly under low-pass filtering. **It subsumes prior work on the topic, provides a theoretical foundation for perceptual optimisation in Monte Carlo rendering, and describes a series of practical algorithms**.

While not directly concerned with sparse mask optimisation, the work aligns with our broader goal of faithful reconstruction – when we interpret the low-pass filter as a reconstruction operator. Here, the optimisation is performed indirectly through the integration sequences, which can be viewed as high-dimensional points in the spatial optimisation framework. In that sense this chapter also combines ideas from inpainting (the rendering process in this case) and denoising (the convolution with a model of the human visual system's point spread function), making it a natural follow-up to Chapter 5.

Finally, a summary and a conclusion is given in Chapter 7, along with an outlook on future work. A list of the figures and the tables, as well as the bibliography and the appendices can be found at the end of the thesis.

Chapter 2

Feature Inpainting

In this chapter, we introduce a general framework for feature-based inpainting that extends the classical inpainting to accommodate arbitrary features expressible as equality constraints. These features can include derivatives, local integrals, and even nonlinear features. The case of linear features combined with a linear inpainting operator, where the inpainting is the gradient of an energy functional, is based on our work [120]. Here, we present a substantial generalisation that supports nonlinear features in conjunction with nonlinear inpainting operators, which are not necessarily derived from an energy functional. This extension is grounded in a theoretical foundation that largely surpasses the scope of [120]. To solve the resulting nonlinear feature inpainting problem, we develop an algorithm inspired by principles from sequential quadratic programming (SQP) [177] and trust region methods [61].

2.1 Outline

We begin with a brief overview of related work in Section 2.2. Next, in Section 2.3.1, we revisit the feature inpainting formulation introduced in our conference paper [120]. We then motivate its extension to cases where the inpainting process is not derived from the gradient of an energy functional – EED inpainting, and also settings where the inpainting matrix $\boldsymbol{L}$ is non-symmetric.

Building on this, we incrementally generalise the feature inpainting framework across several settings:

- Linear inpainting with linear features (Section 2.3.3),
- Nonlinear inpainting with linear features (Section 2.3.4),
- Quasi-linear inpainting with linear features (Section 2.3.5),
- Energy-based nonlinear inpainting with nonlinear features (Section 2.3.6),
- Generic nonlinear inpainting with nonlinear features (Section 2.3.7).

Each section shortly discusses specific solution strategies tailored to its respective setting. For the most general case, we discuss our proposed solver in Section 2.4. We then demonstrate the performance of the theoretical framework through a series of experiments in Section 2.5, focusing on visual comparisons. Efficiency analyses and related experiments are presented in Chapter 3, while detailed MSE comparison tables are included in Chapter 4, as these topics are more appropriately addressed in those chapters. Finally, we conclude the chapter with insights and potential directions for future work in Section 2.6.

2.2 Related Work

Several inpainting-based strategies allocate greyscale or colour interpolation constraints along edges [46, 155] or isolines [216], but these methods are limited to using only pixel intensity or colour values as features. Approaches that combine greyscale data with discontinuity information have been proposed in [112, 118, 119]; however, these rely on segmentation techniques that do not generalise to arbitrary feature types.

Other works have explored image reconstruction from specific features such as scale-space extrema [129], zero-crossings [252], junctions [47], or SIFT descriptors [243]. While these approaches are theoretically insightful, they have not led to practical methods with competitive reconstruction quality. Additionally, KAZE features [7] have been proposed for scale-space analysis, however, they are not used for reconstruction.

More recent methods that incorporate gradient information to improve reconstruction quality include those by Brinkmann et al. [41] and Schneider et al. [211]. However, these approaches also lack generality and do not support the integration of multiple feature types.

Our published work [120] introduces a general and efficient framework for harmonic inpainting with linear features. However, it does not address the extension to nonlinear inpainting operators or nonlinear features. In the current chapter, we present such a generalisation, enabling the use of arbitrary nonlinear features within a nonlinear inpainting framework.

2.3 Feature Inpainting Framework

In the current section we start from the feature inpainting framework described in our conference paper [120], and successively extend it until we arrive at a formulation for nonlinearly constrained nonlinear inpainting.

2.3.1 Linear Constrained Harmonic Inpainting

The discrete harmonic inpainting formulation in Equation (1.4) is limited to interpolating pixel-wise colour data. While this already proves effective in practice, it naturally raises the question of whether more complex features can be incorporated. In [120], we reformulate harmonic inpainting as a constrained optimisation problem:

$$\begin{gathered}\min_u \frac{1}{2}\int_\Omega \|\nabla u(\boldsymbol{x})\|^2\, d\boldsymbol{x} = \min_u \frac{1}{2}\int_\Omega u(\boldsymbol{x})(-\Delta)u(\boldsymbol{x})\, d\boldsymbol{x},\\ \text{subject to } u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathcal{X}.\end{gathered} \tag{2.1}$$

Its discrete counterpart is given by:

$$\min_{\boldsymbol{Cu}=\boldsymbol{Cf}} \frac{1}{2}\|\boldsymbol{Du}\|_2^2 = \min_{\boldsymbol{Cu}=\boldsymbol{Cf}} \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{Lu}. \tag{2.2}$$

In this formulation, the mask matrix $\boldsymbol{C}$ in the constraint $\boldsymbol{Cu} = \boldsymbol{Cf}$ can be replaced by any matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$, enabling interpolation of arbitrary features expressible as linear equality constraints, as demonstrated in [120]. The problem remains linear and can be reformulated by using Lagrange multipliers:

$$\min_{\boldsymbol{Au}=\boldsymbol{Af}} \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{Lu} \Rightarrow \min_{\boldsymbol{u}\in\mathbb{R}^N} \max_{\boldsymbol{\lambda}\in\mathbb{R}^M} \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{Lu} + \boldsymbol{\lambda}^\top \boldsymbol{A}(\boldsymbol{u} - \boldsymbol{f}).$$

The solution satisfies the symmetric but indefinite saddle-point system:

$$\begin{bmatrix} \frac{1}{2}(\boldsymbol{L} + \boldsymbol{L}^\top) & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Af} \end{bmatrix}. \tag{2.3}$$

Uniqueness of the solution is guaranteed if the compression of $\frac{1}{2}(\boldsymbol{L} + \boldsymbol{L}^\top)$ onto the kernel of $\boldsymbol{A}$ is non-singular. For instance the Laplacian matrix $\boldsymbol{L}$ has a kernel spanned by the vector $\boldsymbol{1}$, then if $\boldsymbol{A}\cdot\boldsymbol{1} \neq \boldsymbol{0}$ we get that the system matrix in (2.3) is non-singular. Thus, including an average value constraint or a single pointwise interpolation constraint in $\boldsymbol{A}$ suffices to ensure uniqueness. For symmetric discretisations such as the Laplacian, we have $\frac{1}{2}(\boldsymbol{L} + \boldsymbol{L}^\top) = \boldsymbol{L}$. However, this does not hold for non-symmetric operators. Importantly, this framework can be extended to support operators that are not derived from a minimisation problem, which we explore in the next section.

The Constraint Matrix $\boldsymbol{A}$. As an example, consider matrices representing different features: identity/colours $\boldsymbol{F}_1 = \boldsymbol{I}$, x derivatives $\boldsymbol{F}_2 \approx \partial_x$, y derivatives $\boldsymbol{F}_3 \approx \partial_y$, and local integrals $\boldsymbol{F}_4 \approx K_\sigma *$. To each of those we associate a mask $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_4$. Then the constraints matrix is formed as

$$\boldsymbol{A} = \boldsymbol{CF} = \begin{bmatrix} \boldsymbol{C}_1 & & & \\ & \boldsymbol{C}_2 & & \\ & & \boldsymbol{C}_3 & \\ & & & \boldsymbol{C}_4 \end{bmatrix} \begin{bmatrix} \boldsymbol{F}_1 \\ \boldsymbol{F}_2 \\ \boldsymbol{F}_3 \\ \boldsymbol{F}_4 \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}_1\boldsymbol{F}_1 \\ \boldsymbol{C}_2\boldsymbol{F}_2 \\ \boldsymbol{C}_3\boldsymbol{F}_3 \\ \boldsymbol{C}_4\boldsymbol{F}_4 \end{bmatrix} \in \mathbb{R}^{(4\cdot N)\times N}. \tag{2.4}$$

2.3.2 Compression of Inpainting Operators

An alternative approach to solving the linearly constrained minimisation problem

$$\min_{\boldsymbol{Au}=\boldsymbol{Af}} \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u} \tag{2.5}$$

is to decompose the solution space using orthonormal bases. Let the columns of $\boldsymbol{V}_{\text{ker}} \in \mathbb{R}^{N\times(N-r)}$ form an orthonormal basis for the kernel of $\boldsymbol{A}$, and the columns of $\boldsymbol{V}_{\text{img}} \in \mathbb{R}^{N\times r}$ form an orthonormal basis for its orthogonal complement (i.e., the image of $\boldsymbol{A}^\top$). Then, any solution $\boldsymbol{u}$ can be expressed as

$$\boldsymbol{u} = \boldsymbol{u}_{\text{img}} + \boldsymbol{u}_{\text{ker}} = \boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} + \boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}}. \tag{2.6}$$

Substituting this into the constraint yields

$$\boldsymbol{Au} = \boldsymbol{A}(\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} + \boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}}) = \boldsymbol{A}\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} = \boldsymbol{Af} \quad \Longrightarrow \quad \boldsymbol{s}_{\text{img}} = \boldsymbol{V}_{\text{img}}^\top \boldsymbol{f}. \tag{2.7}$$

Substituting the decomposition into the objective function gives

$$\begin{aligned} E(\boldsymbol{u}) &= \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u} = \frac{1}{2}(\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} + \boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}})^\top \boldsymbol{L}(\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} + \boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}}) \\ &= \frac{1}{2}\boldsymbol{s}_{\text{ker}}^\top \boldsymbol{V}_{\text{ker}}^\top \boldsymbol{L}\boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}} + \boldsymbol{s}_{\text{ker}}^\top \boldsymbol{V}_{\text{ker}}^\top \frac{\boldsymbol{L}+\boldsymbol{L}^\top}{2}\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}} + \frac{1}{2}\boldsymbol{s}_{\text{img}}^\top \boldsymbol{V}_{\text{img}}^\top \boldsymbol{L}\boldsymbol{V}_{\text{img}}\boldsymbol{s}_{\text{img}}. \end{aligned} \tag{2.8}$$

Using $\boldsymbol{s}_{\text{img}} = \boldsymbol{V}_{\text{img}}^\top \boldsymbol{f}$, we minimise the energy with respect to $\boldsymbol{s}_{\text{ker}}$, leading to the linear system

$$\boldsymbol{V}_{\text{ker}}^\top \frac{\boldsymbol{L}+\boldsymbol{L}^\top}{2}\boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}} = -\boldsymbol{V}_{\text{ker}}^\top \frac{\boldsymbol{L}+\boldsymbol{L}^\top}{2}\boldsymbol{V}_{\text{img}}\boldsymbol{V}_{\text{img}}^\top \boldsymbol{f}. \tag{2.9}$$

This system has a unique solution if the compression $\boldsymbol{V}_{\text{ker}}^\top \frac{\boldsymbol{L}+\boldsymbol{L}^\top}{2}\boldsymbol{V}_{\text{ker}}$ is non-singular. While derived in the context of minimisation, this formulation generalises to arbitrary linear operators $\boldsymbol{L} \in \mathbb{R}^{N\times N}$, not necessarily symmetric or positive semi-definite. In such cases, we consider the compressed system

$$\boldsymbol{V}_{\text{ker}}^\top \boldsymbol{L}\boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}} = -\boldsymbol{V}_{\text{ker}}^\top \boldsymbol{L}\boldsymbol{V}_{\text{img}}\boldsymbol{V}_{\text{img}}^\top \boldsymbol{f}. \tag{2.10}$$

Equation (2.10) is equivalent to the augmented system

$$\begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix}\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Af} \end{bmatrix}, \tag{2.11}$$

but it makes explicit that we are solving a problem involving the compression of $\boldsymbol{L}$ onto the kernel of the constraints. This insight also extends to the nonlinear inpainting setting (see Section 2.3.4) and to nonlinear constraints via linearisation (see Section 2.3.7).

2.3.3 Linear Constrained Linear Inpainting

In order to relate (2.10) to the classical formulation

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}, \tag{2.12}$$

we define the orthogonal projection matrix $\boldsymbol{P} = \boldsymbol{V}_{\text{img}}\boldsymbol{V}_{\text{img}}^{\top}$ and proceed as follows:

$$\begin{aligned} \boldsymbol{V}_{\text{ker}}^{\top}\boldsymbol{L}\boldsymbol{V}_{\text{ker}}\boldsymbol{s}_{\text{ker}} &= -\boldsymbol{V}_{\text{ker}}^{\top}\boldsymbol{L}\boldsymbol{V}_{\text{img}}\boldsymbol{V}_{\text{img}}^{\top}\boldsymbol{f}, \\ \boldsymbol{V}_{\text{ker}}\boldsymbol{V}_{\text{ker}}^{\top}\boldsymbol{L}\boldsymbol{V}_{\text{ker}}\boldsymbol{V}_{\text{ker}}^{\top}\boldsymbol{u} &= -\boldsymbol{V}_{\text{ker}}\boldsymbol{V}_{\text{ker}}^{\top}\boldsymbol{L}\boldsymbol{P}\boldsymbol{g}, \\ (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{u} &= -(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}\boldsymbol{P}\boldsymbol{u}, \\ (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})\boldsymbol{u} &= \boldsymbol{P}\boldsymbol{f}. \end{aligned} \tag{2.13}$$

This shows that the inpainting problem with generalised features corresponds to replacing the mask matrix $\boldsymbol{C}$ with the orthogonal projection matrix $\boldsymbol{P}$, which projects onto $\operatorname{span}(\boldsymbol{A}^{\top})$

$$\begin{gathered} (\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f} \quad \Rightarrow \quad (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})\boldsymbol{u} = \boldsymbol{P}\boldsymbol{f}, \\ \text{where } \boldsymbol{P}^2 = \boldsymbol{P}, \quad \boldsymbol{P}^{\top} = \boldsymbol{P}, \quad \operatorname{span}(\boldsymbol{P}) = \operatorname{span}(\boldsymbol{A}^{\top}), \quad \ker(\boldsymbol{P}) = \ker(\boldsymbol{A}). \end{gathered} \tag{2.14}$$

The projection matrix $\boldsymbol{P}$ can be expressed using any $(1,4)$-inverse[1] of $\boldsymbol{A}$ as $\boldsymbol{P} = \boldsymbol{A}^{(1,4)}\boldsymbol{A}$. Alternatively, the formulation can be written as

$$\begin{aligned} (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}\boldsymbol{u} &= \boldsymbol{0}, \\ \boldsymbol{A}\boldsymbol{u} &= \boldsymbol{A}\boldsymbol{f}, \end{aligned} \tag{2.15}$$

analogous to (1.4). This is also equivalent to the augmented system

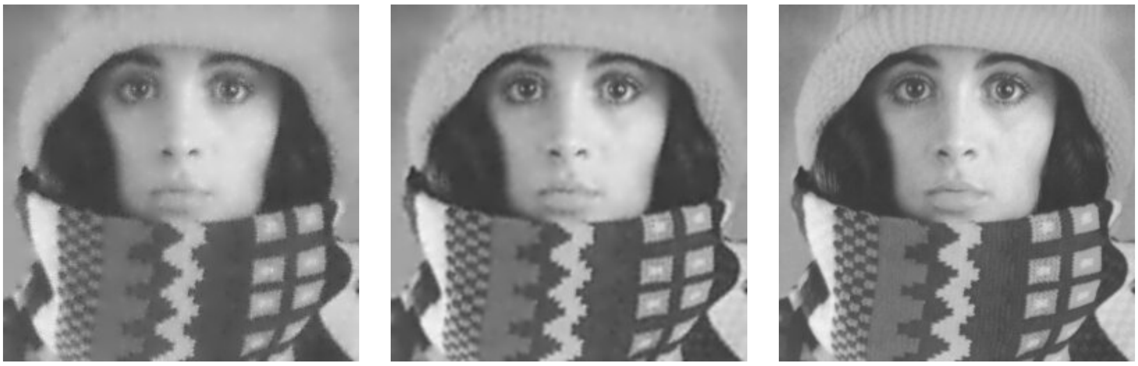
$$\begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^{\top} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{A}\boldsymbol{f} \end{bmatrix}, \tag{2.16}$$

since $\boldsymbol{\lambda}$ can account for any component of $\boldsymbol{L}\boldsymbol{u}$ lying in the span of $\boldsymbol{A}^{\top}$. The augmented system (2.16) is particularly advantageous when computing matrix-vector products with $\boldsymbol{P}$ is computationally expensive. If $\boldsymbol{L}$ is symmetric, one can apply the modified conjugate residual (MCR) method for indefinite systems [52], or the minimal residual (MINRES) method [86, 180]. For non-symmetric matrices $\boldsymbol{L}$, suitable solvers include the stabilised bi-conjugate gradient method (Bi-CGSTAB) [233], the conjugate gradient method applied to the normal equations (CGNR) [32, 105, 206], and related variants such as LSQR [181]. An illustration of this formulation in action is shown in Figure 2.1.

2.3.4 Linear Constrained Nonlinear Inpainting

For a nonlinear inpainting operator $\boldsymbol{\psi} : \mathbb{R}^N \to \mathbb{R}^N$ with linear equality constraints, we can directly extend the previous result by analogy to (1.9) yielding the following inpainting

[1] A $(1,4)$-inverse $\boldsymbol{M}^{(1,4)}$ of $\boldsymbol{M}$ satisfies $\boldsymbol{M}\boldsymbol{M}^{(1,4)}\boldsymbol{M} = \boldsymbol{M}$ and $(\boldsymbol{M}^{(1,4)}\boldsymbol{M})^* = \boldsymbol{M}^{(1,4)}\boldsymbol{M}$.



grey values, MSE: 38.24

five features, MSE: 14.89

original image *trui* $\boldsymbol{f}$

Figure 2.1: Comparison of harmonic inpainting using only grey-value features versus five features (grey values, ∂_x, ∂_y, 3×3 and 5×5 binomial kernels). The mask density is 5% in both cases. **The reconstruction error is more than halved despite using the same inpainting operator and mask density**. Note the significantly improved texture reconstruction of the hat.

formulation

$$\boldsymbol{P u} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{P f}. \tag{2.17}$$

A standard approach to solve this problem is to linearise $\boldsymbol{\psi}$ around a current iterate $\boldsymbol{u}$

$$\boldsymbol{\psi}(\boldsymbol{u} + \boldsymbol{v}) = \boldsymbol{\psi}(\boldsymbol{u}) + \boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u})\boldsymbol{v} + O(\|\boldsymbol{v}\|^2), \tag{2.18}$$

where $\boldsymbol{J}_{\boldsymbol{\psi}}$ denotes the Jacobian of $\boldsymbol{\psi}$. This leads to the following Newton iteration

$$\begin{aligned} \boldsymbol{P u}^{k+1} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\psi}(\boldsymbol{u}^{k+1}) &= \boldsymbol{P f}, \\ \boldsymbol{P}(\boldsymbol{u}^k + \boldsymbol{v}^k) + (\boldsymbol{I} - \boldsymbol{P})(\boldsymbol{\psi}(\boldsymbol{u}^k) + \boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k)\boldsymbol{v}^k) &= \boldsymbol{P f}, \\ (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k))\boldsymbol{v}^k &= \boldsymbol{P f} - \boldsymbol{P u}^k - (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\psi}(\boldsymbol{u}^k). \end{aligned} \tag{2.19}$$

The corresponding augmented system is

$$\begin{bmatrix} \boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k) & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^k \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\psi}(\boldsymbol{u}^k) \\ \boldsymbol{A f} - \boldsymbol{A u}^k \end{bmatrix}. \tag{2.20}$$

A potential issue arises if the compression of $\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k)$ on the kernel of $\boldsymbol{A}$ is singular. In such cases, the system may be inconsistent, i.e., $(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\psi}(\boldsymbol{u}^k)$ may not lie in the range of $(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k)$. In practice, one can project $\boldsymbol{\psi}(\boldsymbol{u}^k)$ onto span$(\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k))$, e.g., using CGNR, and adjust the right-hand side of (2.20) accordingly.

Alternatively, if the augmented system (2.20) is symmetric, one obtains the pseudoinverse solution automatically by applying the conjugate residual (CR) solver [86] or the MINRES method [86, 180], with the modification proposed in [150]. For non-symmetric systems, the conjugate gradient method for the normal equations (CGNR) [32, 105, 206], initialised with a vector orthogonal to the kernel (e.g., the zero vector), also yields the pseudoinverse solution [101].

It is important to note that in this formulation, $\boldsymbol{\psi}$ need not be the gradient of an energy functional. The method directly linearises the inpainting equation

$$\boldsymbol{P}\boldsymbol{u} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{P}\boldsymbol{f},$$

making it applicable to inpainting operators such as EED, which lack an energy [244].

Within this framework, computing the Jacobian $\boldsymbol{J}_{\boldsymbol{\psi}}$ of the inpainting operator is essential. While automatic differentiation makes this feasible, it can be computationally intensive and may yield a non-symmetric Jacobian if $\boldsymbol{\psi} \neq \nabla E$ for some energy functional E. In Section 2.3.5, we demonstrate a simplification for quasi-linear inpainting operators (such as EED), which take the form $\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{L}(\boldsymbol{u})\boldsymbol{u}$. This avoids the explicit computation of the Jacobian and – if $\boldsymbol{L}(\boldsymbol{u})$ is symmetric – results in a symmetric system.

2.3.5 Linear Constrained Quasi-Linear Inpainting

If the inpainting operator has the form $\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{L}(\boldsymbol{u})\boldsymbol{u}$, we can avoid computing the full Jacobian by employing a Kačanov iteration [102, 123] instead of a Newton iteration. The Kačanov method effectively omits the term involving $\boldsymbol{J}_{\boldsymbol{L}}$ from the Jacobian $\boldsymbol{J}_{\boldsymbol{\psi}}$:

$$\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}) = \boldsymbol{L}(\boldsymbol{u}) + \boldsymbol{J}_{\boldsymbol{L}}(\boldsymbol{u})\boldsymbol{u} \approx \boldsymbol{L}(\boldsymbol{u}). \tag{2.21}$$

This approximation is valid in practice when $\boldsymbol{J}_{\boldsymbol{L}}$ does not vary too rapidly with $\boldsymbol{u}$. For instance, in EED inpainting, this corresponds to settings where the contrast parameter λ is not too small.

Under this approximation, the inpainting formulation simplifies to

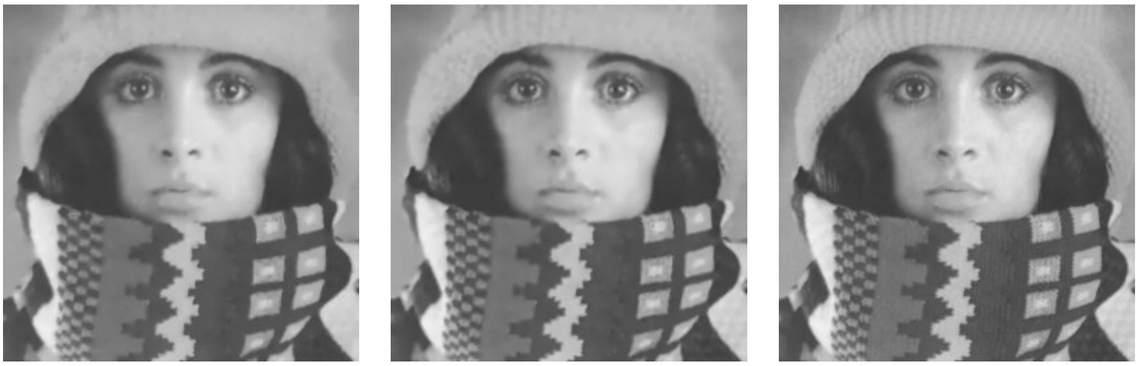
$$(\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{u}^k))\boldsymbol{u}^{k+1} = \boldsymbol{P}\boldsymbol{f}, \tag{2.22}$$

with the corresponding augmented system

$$\begin{bmatrix} \boldsymbol{L}(\boldsymbol{u}^k) & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{k+1} \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{A}\boldsymbol{f} \end{bmatrix}. \tag{2.23}$$

This approach offers several advantages. If $\boldsymbol{L}(\boldsymbol{u}^k)$ is symmetric – as is the case for EED – the system remains symmetric, whereas the full Jacobian $\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}) = \boldsymbol{L}(\boldsymbol{u}) + \boldsymbol{J}_{\boldsymbol{L}}(\boldsymbol{u})\boldsymbol{u}$ is generally not symmetric due to the second term. Additionally, this formulation avoids the potential inconsistency that may arise when projecting $\boldsymbol{J}_{\boldsymbol{\psi}}$ onto the kernel of $\boldsymbol{A}$, since the right-hand side is zero. As a result, solvers for consistent systems can be employed, such as projected conjugate gradients or SYMMLQ [180] in the symmetric case, and Bi-CGSTAB [233] in the non-symmetric case.

An illustration of the results obtained by using this formulation is shown in Figure 2.2.



grey values, MSE: 14.95 five features, MSE: 10.77 original image *trui* $\boldsymbol{f}$

FIGURE 2.2: Illustration of EED inpainting using only grey-value features versus five features (grey values, ∂_x, ∂_y, 3×3 and 5×5 binomial kernels). The total mask density is 5% in both cases. **The reconstruction error improves by nearly 28% compared to the already strong performance of classical EED inpainting**.

2.3.6 Optimisation Formulation for Nonlinear Features

Here, we motivate an inpainting formulation with nonlinear features from an optimisation perspective, analogous to the approach in Section 2.3.1. This serves as a precursor to the formulation in the next section, which does not rely on an optimisation problem.

Let $\boldsymbol{b} : \mathbb{R}^N \to \mathbb{R}^M$ be a C^2 nonlinear function representing the constraints, and let the inpainting operator $\boldsymbol{\psi}(\boldsymbol{u})$ be the gradient of a C^2 energy functional $E : \mathbb{R}^N \to \mathbb{R}$, i.e., $\boldsymbol{\psi}(\boldsymbol{u}) = \nabla E(\boldsymbol{u})$. The inpainting task can then be formulated as an equality-constrained optimisation problem:

$$\min_{\boldsymbol{b}(\boldsymbol{u})=0} E(\boldsymbol{u}) \quad \Longrightarrow \quad \min_{\boldsymbol{u}\in\mathbb{R}^N} \max_{\boldsymbol{\lambda}\in\mathbb{R}^M} \mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}), \quad \mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) := E(\boldsymbol{u}) + \boldsymbol{\lambda}^\top \boldsymbol{b}(\boldsymbol{u}). \tag{2.24}$$

A stationary point of this saddle-point problem must satisfy the first-order optimality conditions:

$$\nabla_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = \nabla E(\boldsymbol{u}) + \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u})\boldsymbol{\lambda} = \boldsymbol{0}, \tag{2.25}$$

$$\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = \boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{0}. \tag{2.26}$$

Linearising around the current iterate $\boldsymbol{u}^k$ yields the following augmented system:

$$\begin{bmatrix} \nabla_{\boldsymbol{u}\boldsymbol{u}} \mathcal{L}(\boldsymbol{u}^k,\boldsymbol{\lambda}^k) & \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u}^k) \\ \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^k \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = - \begin{bmatrix} \nabla E(\boldsymbol{u}^k) \\ \boldsymbol{b}(\boldsymbol{u}^k) \end{bmatrix}. \tag{2.27}$$

In this formulation, we have:

$$\nabla E(\boldsymbol{u}) = \boldsymbol{\psi}(\boldsymbol{u}), \quad \nabla_{\boldsymbol{u}\boldsymbol{u}} \mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = \boldsymbol{J}_{\psi}(\boldsymbol{u}) + \boldsymbol{H}[\boldsymbol{b}](\boldsymbol{u})\boldsymbol{\lambda} = \boldsymbol{J}_{\psi}(\boldsymbol{u}) + \sum_{j=1}^{M} \boldsymbol{H}[b_j](\boldsymbol{u})\lambda_j, \tag{2.28}$$

where $\boldsymbol{H}[b_j](\boldsymbol{u})$ denotes the Hessian of the j-th constraint function b_j from the constraint vector $\boldsymbol{b}$. The system in (2.27) corresponds to the standard sub-problem solved at each iteration of sequential quadratic programming (SQP) [177].

2.3.7 Nonlinear System Formulation for Nonlinear Features

We now generalise the previous formulation by dropping the assumption that $\boldsymbol{\psi} = \nabla E$. Instead, we begin directly from the stationarity conditions (2.25) and (2.26), which now take the form:

$$\boldsymbol{\psi}(\boldsymbol{u}) + \boldsymbol{J}_{\boldsymbol{b}}^{\top}(\boldsymbol{u})\boldsymbol{\lambda} = \boldsymbol{0}, \tag{2.29}$$

$$\boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{0}. \tag{2.30}$$

The above system is equivalent to:

$$(\boldsymbol{I} - \boldsymbol{P}(\boldsymbol{u}))\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{0}, \tag{2.31}$$

$$\boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{0}, \tag{2.32}$$

where $\boldsymbol{I} - \boldsymbol{P}(\boldsymbol{u})$ denotes the orthogonal projector onto the kernel of the Jacobian $\ker(\boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}))$ of the constraints. This generalises the linear formulation in (2.15), which is recovered by setting $\boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{A}\boldsymbol{f}$ and $\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{L}\boldsymbol{u}$.

Equations (2.31) and (2.32) thus define the general nonlinear inpainting formulation with nonlinear equality constraints: we enforce the constraints and require that $\boldsymbol{\psi}(\boldsymbol{u})$ lies in the kernel of $\boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u})$. Linearising this system yields the compressed Jacobian $(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{I} - \boldsymbol{P})$.

For large and sparse problems, it is often more practical to work with the augmented system:

$$\begin{bmatrix} \boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k) + \boldsymbol{H}[\boldsymbol{b}](\boldsymbol{u}^k)\boldsymbol{\lambda}^k & \boldsymbol{J}_{\boldsymbol{b}}^{\top}(\boldsymbol{u}^k) \\ \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^k \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{\psi}(\boldsymbol{u}^k) \\ \boldsymbol{b}(\boldsymbol{u}^k) \end{bmatrix}. \tag{2.33}$$

In this system, it may happen that neither $-\boldsymbol{\psi}(\boldsymbol{u}^k)$ nor $-\boldsymbol{b}(\boldsymbol{u}^k)$ lies in the range of the system matrix. While one could apply a least-squares solver such as CGNR directly, in practice it is more consequent to project $\boldsymbol{\psi}(\boldsymbol{u}^k)$ onto $\operatorname{span}(\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k) + \boldsymbol{H}[\boldsymbol{b}](\boldsymbol{u}^k)\boldsymbol{\lambda}^k)$ and $\boldsymbol{b}(\boldsymbol{u}^k)$ onto $\operatorname{span}(\boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k))$, and then modify the right-hand side accordingly. Based on our experience, this projection step can be crucial for obtaining a meaningful search direction in each SQP iteration.

In the quasi-linear setting, where $\boldsymbol{\psi}(\boldsymbol{u}) = \boldsymbol{L}(\boldsymbol{u})\boldsymbol{u}$ and the approach from Section 2.3.5 is used, inconsistency in the system cannot arise unless the term $\boldsymbol{H}[\boldsymbol{b}]\boldsymbol{\lambda}^k$ reduces the rank of $\boldsymbol{L}(\boldsymbol{u}^k)$. However, if both $\boldsymbol{L}(\boldsymbol{u}^k)$ and $\boldsymbol{H}[\boldsymbol{b}](\boldsymbol{u}^k)\boldsymbol{\lambda}^k$ are positive semi-definite, this cannot occur, since for positive semi-definite matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, we have:

$$\operatorname{rank}(\boldsymbol{A} + \boldsymbol{B}) \geq \max(\operatorname{rank}(\boldsymbol{A}), \operatorname{rank}(\boldsymbol{B})).$$

The Constraint Functions $\boldsymbol{b}$. As an example, consider functions representing different linear or nonlinear features: $\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_k : \mathbb{R}^N \to \mathbb{R}^N$, and their associated masks $\boldsymbol{c}_1, \dots, \boldsymbol{c}_k$.

Then we can form $\boldsymbol{b}$ as follows:

$$\boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{u}) - \boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{f}) = \begin{bmatrix} \boldsymbol{C}_1\big(\boldsymbol{\phi}_1(\boldsymbol{u}) - \boldsymbol{\phi}_1(\boldsymbol{f})\big) \\ \vdots \\ \boldsymbol{C}_k\big(\boldsymbol{\phi}_k(\boldsymbol{u}) - \boldsymbol{\phi}_k(\boldsymbol{f})\big) \end{bmatrix} \in \mathbb{R}^{(k\cdot N)\times N}. \tag{2.34}$$

In the linear setting we have that $\boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{u}) = \boldsymbol{C}\boldsymbol{F}\boldsymbol{u} = \boldsymbol{A}\boldsymbol{u}$, and thus $\boldsymbol{b}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{A}\boldsymbol{f}$.

2.4 Solution Strategy

In the setting of linear constraints with a quasi-linear inpainting operator, as discussed in Section 2.3.5, solving the inpainting problem is relatively straightforward. A few Kačanov iterations (possibly damped [102]) typically suffice, and the resulting linear systems (2.23) can be efficiently solved using, for example, the conjugate residual method. In contrast, nonlinear constraints introduce significant complexity, as the normal and tangent spaces of the constraints' "manifold"[2] vary from iteration to iteration. Below we briefly describe our SQP-inspired algorithm for the nonlinear feature inpainting. Standard SQP algorithms [61, 143] are applicable only to the optimisation formulation, and for large problems they use a projected conjugate gradient solver [143]. From our experiments we have seen that our algorithm is much more robust w.r.t. convergence to the solution compared to standard SQP, even in the energy-derived inpainting setting where classical SQP is still applicable.

Line Search vs Trust Region. Newton-type methods are not globally convergent in general. To address this, globalisation strategies such as line search or trust region methods are commonly employed [61, 177]. In our setting, we observed that line-search-based SQP methods often fail to converge without additional regularisation, even when using theoretically sound acceptance criteria [45]. This failure appears to stem partly from rank deficiency in $\boldsymbol{J_b}$ and partly from the Newton directions becoming nearly orthogonal to the gradient in ill-conditioned problems [230]. Forcing sequences that relax solver tolerances [79] can help, but we have still experienced cases where the method failed to converge. As such we focus on trust region approaches, which have proven to be much more robust in our setting.

Byrd-Omojokun Step Decomposition. Trust region methods using Steihaug-Krylov solvers [221] tend to be more robust in our problem setting, though they may still stall. A key component is the quasi-normal step from the Byrd-Omojokun trust region framework (see [143, 177]), which provides a reliable direction by solving:

$$\min_{\boldsymbol{v}_N^k \in \mathcal{C}^k} \|\boldsymbol{v}_N^k\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{v}_N^k\|_2 \le \xi \Delta_k, \quad \mathcal{C}^k = \arg\min_{\boldsymbol{v}} \|\boldsymbol{b}(\boldsymbol{u}^k) + \boldsymbol{J_b}(\boldsymbol{u}^k)\boldsymbol{v}\|_2^2, \tag{2.35}$$

[2] We allow for the tangent spaces degenerating, i.e., the constraints' hyper-surface does not need to be regular.

yielding the quasi-normal direction $\boldsymbol{v}_N^k$. The full step is given by $\boldsymbol{v}_N^k = -\boldsymbol{J}_{\boldsymbol{b}}^+(\boldsymbol{u}^k)\boldsymbol{b}(\boldsymbol{u}^k)$, but the trust region constraint $\xi\Delta_k$ may necessitate truncation. Once feasibility is improved, a tangential step $\boldsymbol{v}_T^k$ is computed to reduce the objective within the tangent space:

$$\begin{bmatrix} \boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k) + \boldsymbol{H}[\boldsymbol{b}]\boldsymbol{\lambda}^k & \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u}^k) \\ \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_N^k + \boldsymbol{v}_T^k \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{\psi}(\boldsymbol{u}^k) \\ \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k)\boldsymbol{v}_N^k \end{bmatrix}, \text{ with } \|\boldsymbol{v}_N^k + \boldsymbol{v}_T^k\|_2 \leq \Delta_k. \quad (2.36)$$

This formulation ensures consistency with respect to the normality conditions.

Model and Merit Functions. In practice, both trust region constraints[3] are enforced using a Steihaug-CGNR solver, which terminates when the step size exceeds the trust region radius. Since CGNR monotonically reduces the residual norm, we define the model function as:

$$m_k(\boldsymbol{v}) = \min_{\boldsymbol{\lambda}} \|(\boldsymbol{J}_{\boldsymbol{\psi}}(\boldsymbol{u}^k) + \boldsymbol{H}[\boldsymbol{b}])\boldsymbol{v} + \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u}^k)\boldsymbol{\lambda}\|_2^2 + \rho^2\|\boldsymbol{b}(\boldsymbol{u}^k) + \boldsymbol{J}_{\boldsymbol{b}}(\boldsymbol{u}^k)\boldsymbol{v}\|_2^2. \quad (2.37)$$

The corresponding merit function is based on the residuals of the original nonlinear system (2.29)–(2.30):

$$\varphi_k(\boldsymbol{v}) = \min_{\boldsymbol{\lambda}} \|\boldsymbol{\psi}(\boldsymbol{u}^k + \boldsymbol{v}) + \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u}^k + \boldsymbol{v})\boldsymbol{\lambda}\|_2^2 + \rho^2\|\boldsymbol{b}(\boldsymbol{u}^k + \boldsymbol{v})\|_2^2. \quad (2.38)$$

In practice, we omit the minimisation over $\boldsymbol{\lambda}$ and substitute $\boldsymbol{\lambda}^{k+1}$ in $m_k(\boldsymbol{v}^k)$ and $\varphi_k(\boldsymbol{v}^k)$, and $\boldsymbol{\lambda}^k$ in $m_k(\boldsymbol{0}) = \varphi_k(\boldsymbol{0})$. Empirically, $\boldsymbol{\lambda}^k$ converges toward the optimal multiplier as the method progresses. The weight ρ can either remain fixed throughout or it can be updated to more quickly enforce the constraints at the expense of the objective function. Note that ξ should ideally be a function of ρ.

Trust Region Update. Step acceptance and trust region updates follow standard rules [61, 177], based on the ratio of actual to predicted reduction:

$$r_k = \frac{ared}{pred} = \frac{\varphi_k(\boldsymbol{0}) - \varphi_k(\boldsymbol{v}^k)}{m_k(\boldsymbol{0}) - m_k(\boldsymbol{v}^k)} \quad \Rightarrow \quad \Delta_{k+1} \in \begin{cases} \gamma_2\Delta_k, & \text{if } r_k \geq \eta_2 \text{ and } \|\boldsymbol{v}^k\| = \Delta_k, \\ \Delta_k, & \text{if } r_k \in [\eta_1, \eta_2), \\ \gamma_1\Delta_k, & \text{if } r_k < \eta_1. \end{cases} \quad (2.39)$$

Here, $0 < \eta_1 \leq \eta_2 \leq 1$ are thresholds for rejecting a step ($r_k < \eta_1$), accepting it without changing the trust region ($r_k \in [\eta_1, \eta_2)$), or accepting it and expanding the region ($r_k \geq \eta_2$), provided the step was constrained by the trust region. We have experimented with both the practical parameters suggested in [61] and [177]. From our empirical experience the parameters from [61] yield slightly faster convergence, however, the variation is not very large.

[3] The trust region constraints are: $\|\boldsymbol{v}_N^k\|_2 \leq \xi\Delta_k$ from (2.35), and $\|\boldsymbol{v}_N^k + \boldsymbol{v}_T^k\|_2 \leq \Delta_k$ from (2.36).

2.5 Experiments

To illustrate the inpainting approach, we present a selection of representative results without delving into extensive detail. A more thorough experimental evaluation is deferred to the data optimisation chapter (Chapter 4), where it is more appropriately situated in the context of algorithms designed for optimising reconstruction data.

2.5.1 Experimental Setup

Linear Features. We define five types of linear features, each expressible via boundary-reflecting convolutions:

$$(\boldsymbol{\phi}_1(\boldsymbol{v}))_{i,j} = v_{i,j}, \quad (\boldsymbol{\phi}_2(\boldsymbol{v}))_{i,j} = v_{i+1,j} - v_{i,j}, \quad (\boldsymbol{\phi}_3(\boldsymbol{v}))_{i,j} = v_{i,j+1} - v_{i,j},$$
$$\boldsymbol{\phi}_4(\boldsymbol{v}) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 2 & 4 & 2 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \boldsymbol{v}, \quad \boldsymbol{\phi}_5(\boldsymbol{v}) = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} * \boldsymbol{v}. \tag{2.40}$$

Here, $\boldsymbol{\phi}_1$ is the identity feature used for classical colour interpolation, $\boldsymbol{\phi}_2$ and $\boldsymbol{\phi}_3$ represent discrete derivatives, and $\boldsymbol{\phi}_4$ and $\boldsymbol{\phi}_5$ approximate Gaussian smoothing. Stencil normalisation is handled automatically by our spatial optimisation in Chapter 4, so multiplicative constants do not have any effect.

Nonlinear Feature. In the nonlinear setting, we replace $\boldsymbol{\phi}_5$ with a weighted variance feature:

$$(\tilde{\boldsymbol{\phi}}_4)_{i,j}(\boldsymbol{u}) = \frac{1}{2}\sum_{k=-1}^{1}\sum_{l=-1}^{1} w_{kl}(u_{i+k,j+l} - \mu_{i,j})^2, \quad \mu_{i,j} = \sum_{k=-1}^{1}\sum_{l=-1}^{1} w_{kl}u_{i+k,j+l}, \tag{2.41}$$

where the weighting stencil is the normalised version of the one used in $\boldsymbol{\phi}_4$:

$$\boldsymbol{W} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{2.42}$$

Harmonic Inpainting. For harmonic inpainting, we use the standard 5-point stencil discretisation of the negated Laplacian:

$$(-\Delta u)(x_i, y_j) \approx (\boldsymbol{L}\boldsymbol{u})_{i,j} = \frac{-u_{i-1,j} + 2u_{i,j} - u_{i+1,j}}{h_x^2} + \frac{-u_{i,j-1} + 2u_{i,j} - u_{i,j+1}}{h_y^2}, \tag{2.43}$$

with reflecting boundary conditions: $u_{0,j} := u_{1,j}$, $u_{W+1,j} := u_{W,j}$, $u_{i,0} := u_{i,1}$, $u_{i,H+1} := u_{i,H}$.

EED Inpainting. The matrix for EED inpainting is constructed following the discretisation in [241]. We use the following parameters: contrast parameter $\lambda = 1.0$, Gaussian pre-smoothing standard deviation $\sigma = 0.8$, and parameters $\alpha = 0.25$, $\gamma = 1.0$ (for more details on those see [241]). The EED inpainting process is initialised with the harmonic inpainting result, followed by Kačanov iterations to compute the final solution.

2.5.2 Visual Comparisons

We provide visual comparisons for harmonic and EED inpainting that illustrate results from our generalised inpainting framework. Results such as mean-squared-errors and the generated masks for the different features are discussed in Chapter 4. We also illustrate an inpainting with the nonlinear feature in Fig. 2.5, but unfortunately it does not produce meaningful improvements, while the inpainting is many times more computationally expensive. Consequently, the selection of good nonlinear features remains an open problem.

Harmonic Inpainting. Figure 2.3 showcases the large difference between harmonic inpainting with only colour interpolation and with the five linear features. We note that the classical harmonic inpainting is both much blurrier and suffers from pronounced logarithmic singularities. More visual comparisons can be found in Fig. 2.6. We highly recommend zooming into the images on a computer in order to better appreciate the difference. The differences are very obvious if one switches between the images on a computer.

EED Inpainting. Figure 2.4 illustrates the visual difference between EED inpainting with only colour interpolation and with the five features. The latter is less blurry and wavy and has closed some image edges better. Additional visual comparisons can be found in Fig. 2.7.

2.6 Conclusion

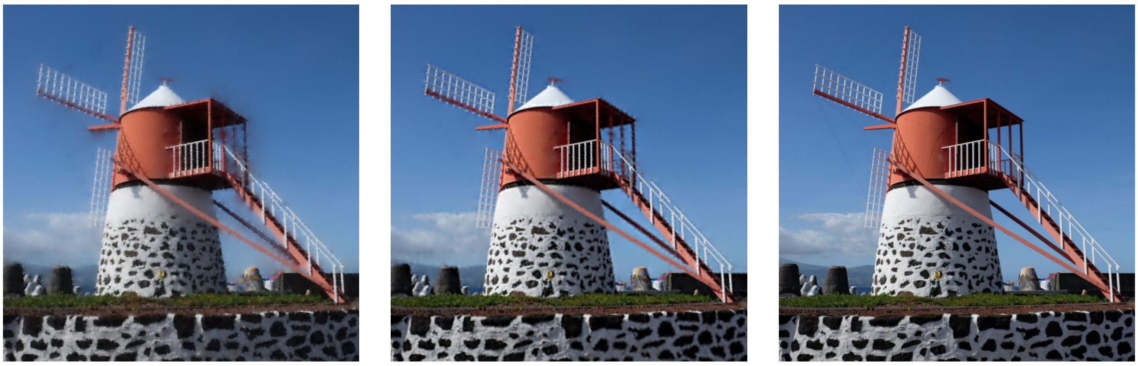
We have introduced a general theoretical framework for nonlinear inpainting with nonlinear equality constraints. The key insight is that, unlike the classical formulation, it is sufficient to replace the mask matrix $\boldsymbol{C}$ with the orthogonal projection matrix $\boldsymbol{P} = \boldsymbol{J}_{\boldsymbol{b}}^{+}\boldsymbol{J}_{\boldsymbol{b}}$, which rejects the kernel of the constraints' Jacobian. This formulation aligns with optimisation-based approaches and remains applicable even when the inpainting operator is not derived from an energy functional.

Our theory highlights an important conceptual point: Prescribing a set of features inherently alters the inpainting operator. The problem being solved involves the compression of the operator onto the kernel of the linearised constraints – thus, feature selection implicitly defines the behaviour of the reconstruction process beyond just the pointwise interpolation of the features.

A key strength of our formulation lies in its flexibility: it enables the seamless integration of diverse features. Among these, we have identified colours, derivatives, and local integrals as particularly effective, leading to significant improvements in reconstruction quality.

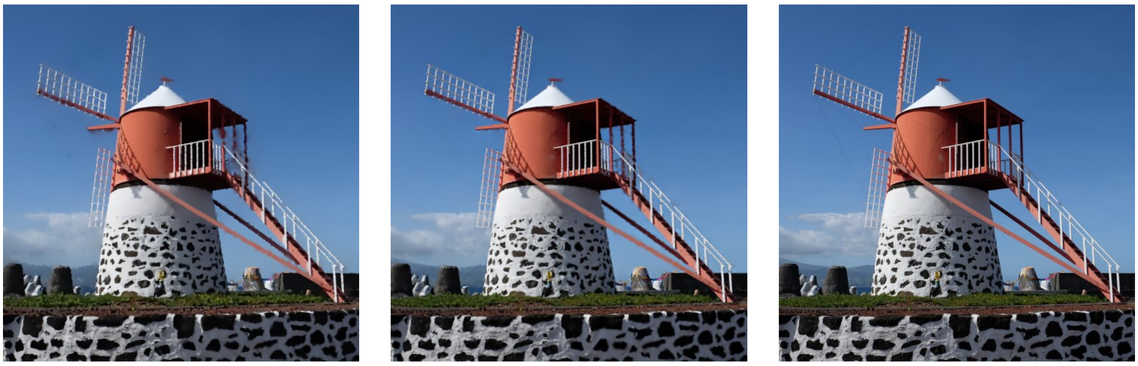
While we have discussed various solution strategies, we have not yet addressed computational efficiency – this topic is partially explored in Chapter 3. Likewise, we have not tackled the problem of optimally allocating feature types ratios, selecting interpolation point locations, or determining the best coefficients at those points. These aspects are the focus of Chapter 4.

Finally, we have not delved into a detailed study of practically relevant nonlinear features, which remains an open direction for future research. Another promising avenue is the extension of our framework to accommodate inequality-constrained features. The theoretical foundation for this is straightforward, relying on the Karush-Kuhn-Tucker (KKT) conditions resulting from an inequality-constrained optimisation problem. However, solving such problems will require more sophisticated techniques, such as inequality-constrained SQP-inspired methods or interior-point approaches. In terms of practically relevant inequality-constrained features, one could, for example, enforce local minima and maxima in order to prevent overshoots and undershoots – potentially a useful modification for biharmonic inpainting.



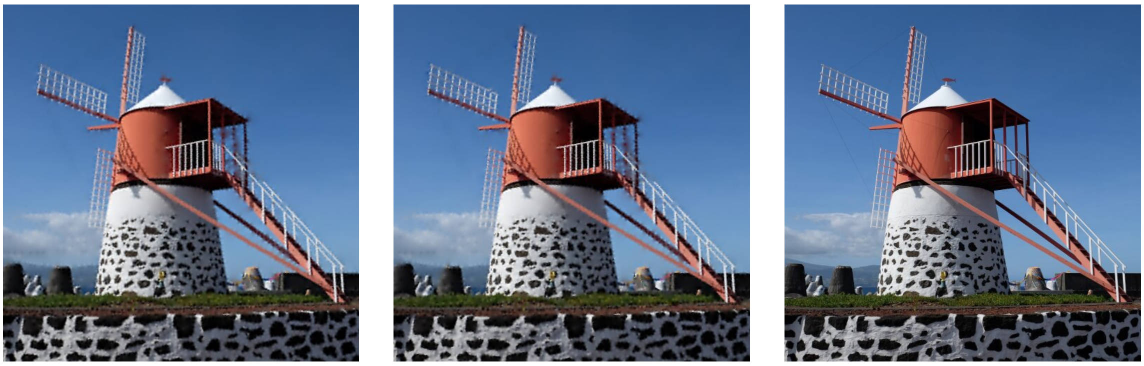
colour values only all five features original

Figure 2.3: Harmonic inpainting with linear features, 5% mask. Note the large improvements in the closing of the edges and the suppression of the logarithmic singularities inherent to harmonic inpainting.



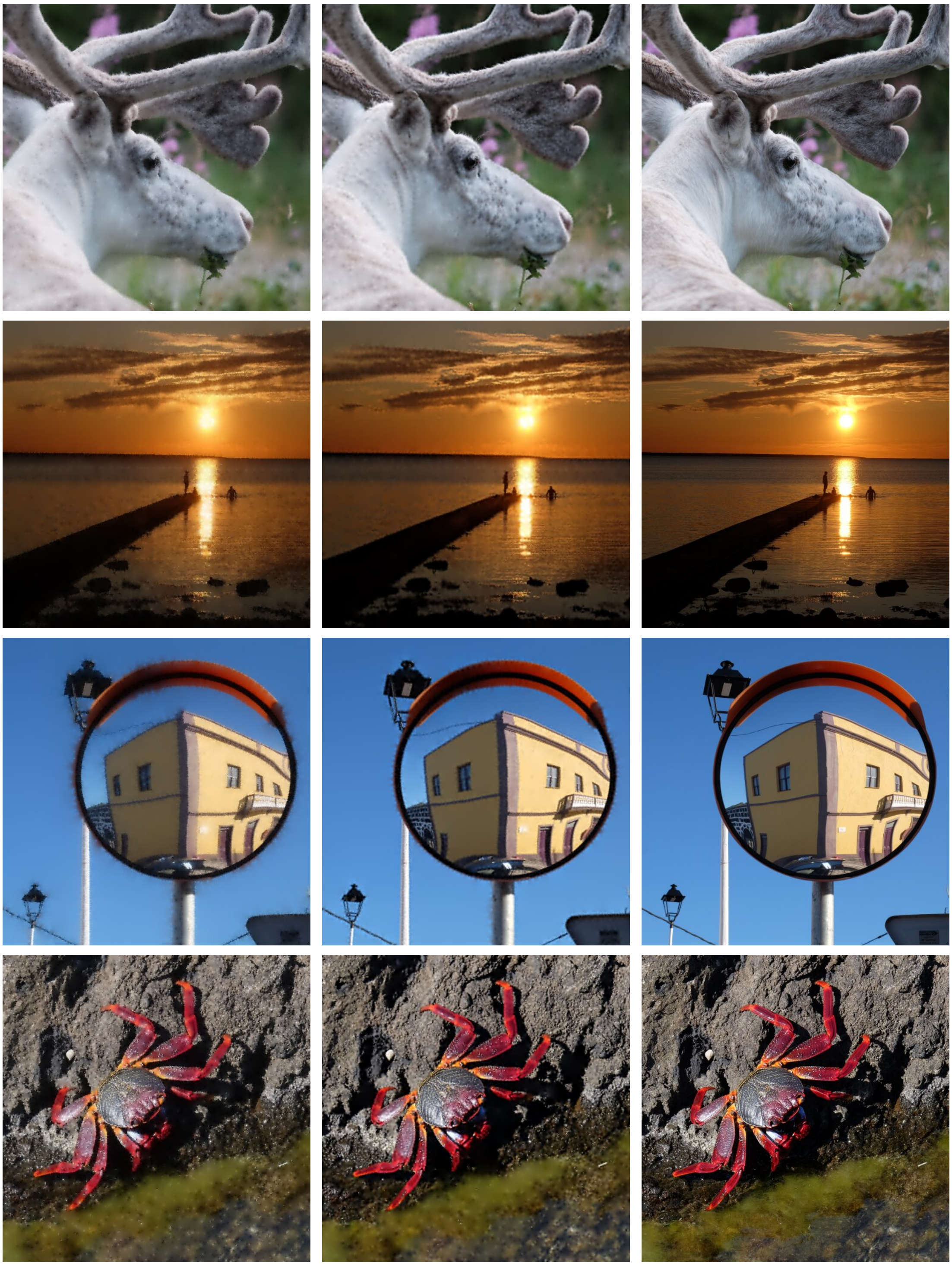
colour values only all five features original

Figure 2.4: EED inpainting with linear features, 5% mask. Note the much sharper result and better reconstructed edges.



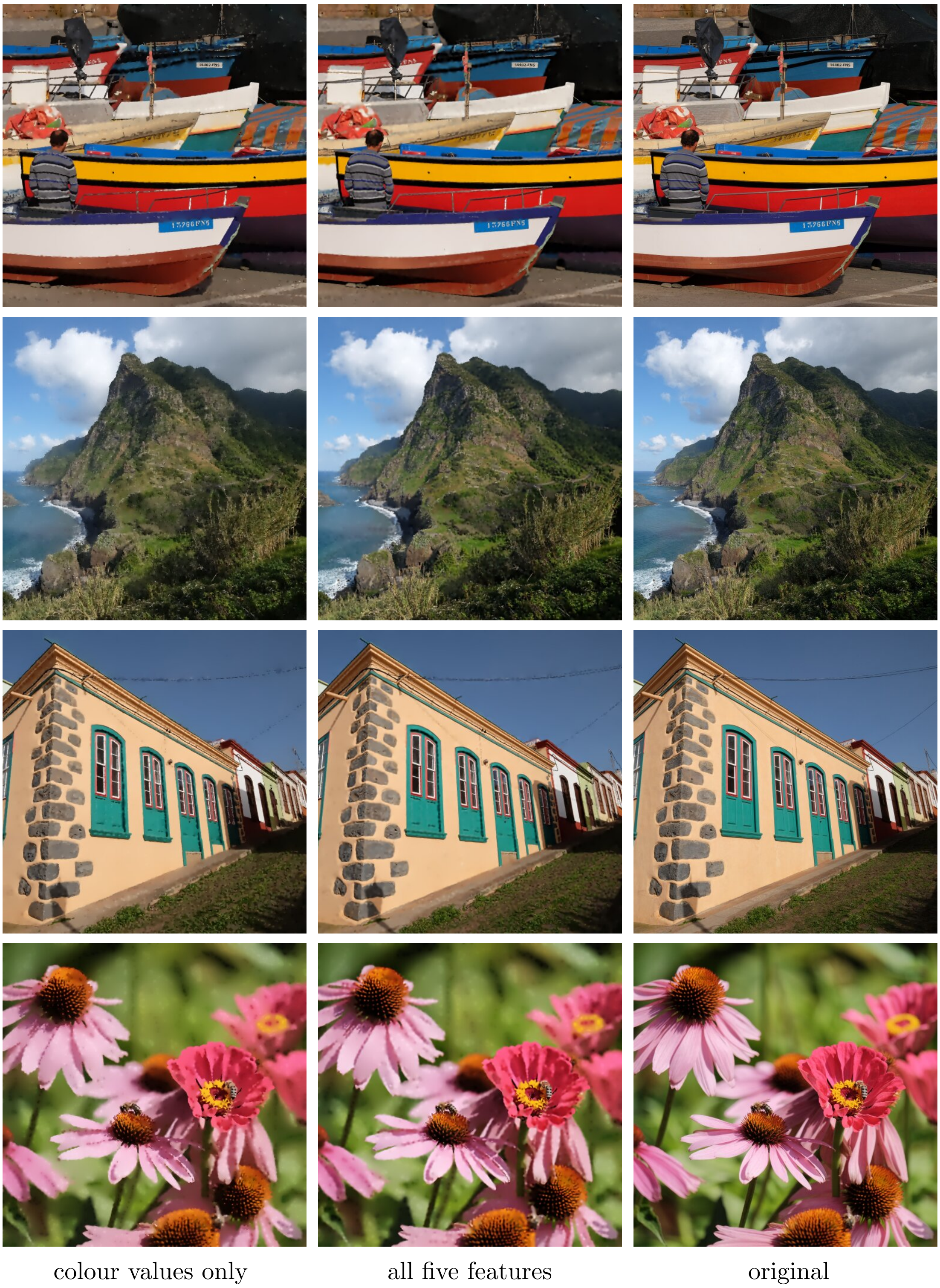
lin. constr., MSE: 110.19 nonlin. constr., MSE: 108.24 original

Figure 2.5: *windmill* with four linear features (left) vs four linear features and the nonlinear weighted variance feature (middle), 5% mask. Unfortunately the nonlinear feature brings no tangible benefits.



colour values only all five features original

Figure 2.6: Harmonic inpainting with the same total mask density. Photos by Joachim Weickert – top to bottom: *raindeer, quai, mirror, crab.* Note the much sharper *mirror* image reconstruction.



colour values only

all five features

original

Figure 2.7: EED inpainting with the same total mask density. Photos by Joachim Weickert – top to bottom: *boats, madeira, garafia, flowers.*

Chapter 3

Fast Inpainting Methods

Three of our works [58, 127, 128] discuss strategies for accelerating image inpainting. Section 3.1 is based on our work [58], where we employ the finite element method (FEM) [117, 144] to formulate inpainting on an adaptive triangular mesh. This approach significantly reduces the system size, enabling a notable speed-up compared to standard finite difference method (FDM) [168] implementations. Additionally, the use of a coarser FEM mesh helps suppress the logarithmic singularities that are characteristic of harmonic inpainting in two dimensions. We also discuss extensions to polyharmonic inpainting that are not present in [58] but that we have presented at GAMM23.

In contrast, in Section 3.2 we reproduce parts of our work [127, 128] on fast harmonic inpainting in the FDM setting. There we target GPU architectures, leveraging multigrid and domain decomposition techniques to achieve real-time inpainting performance on 4K-resolution images. We focus on our main contribution there related to the multigrid implementation.

3.1 FEM for Harmonic Inpainting

Finite difference methods (FDM) [168] and finite element methods (FEM) [117, 144] are two classical numerical approaches for solving differential equations. FDM is typically implemented on regular, equidistant grids – such as the pixel grid of digital images – while FEM is particularly well-suited for adaptive meshes. As a result, the number of unknowns in FDM grows rapidly with image resolution, whereas FEM allows fine control over the number of unknowns through the mesh construction process. This flexibility naturally motivates the use of an adaptive, lower-resolution FEM mesh to accelerate inpainting compared to FDM.

In the context of FDM, a comparable idea would be to downsample the image. However, this would result in a resolution that is too coarse in some regions and unnecessarily fine in others, since classical FDM does not support adaptivity.

3.1.1 Outline

We discuss related work in Section 3.1.2, after which we give an extremely brief description of the finite element method (FEM) in Section 3.1.3, followed by the formulation of harmonic inpainting in the finite element setting in Section 3.1.4. Specifics of our implementation are discussed in Section 3.1.5. We illustrate our approach with some experiments in Section 3.1.6. We showcase extensions that go beyond our work [58] in Section 3.1.7, and present our preferred formulation for polyharmonic inpainting in Section 3.1.8. We conclude the finite element section of this chapter in Section 3.1.9, where we also discuss avenues for future research.

3.1.2 Related Work

Finite element methods (FEM) have been successfully applied to PDE-based models for image denoising [20, 133, 196] and restoration [37, 226]. However, to the best of our knowledge, they had not been used for PDE-based image approximation from sparse data prior to our work [58]. They have been used for surface reconstruction, however, in the work of Bae and Weickert [19]. Additionally, since the publication of our work, an FEM inpainting-based approach has been used for MRI data [122].

3.1.3 Brief Description of FEM

Unlike the finite difference method, which fits polynomials and takes their pointwise derivatives, the finite element method (FEM) relies on integral forms of the differential equations by integrating against test functions $v \in \mathcal{V}$. After this, the formulation is typically brought to a weak form where the highest order of the derivatives is reduced in half. This step can be skipped, but it is typically done because it allows one to use simpler subspaces in the discretisation, and allows for weak solutions. Finally, one discretises the resulting equations by considering finite-dimensional subspaces $\mathcal{V}_h \subset \mathcal{V}$, $\mathcal{U}_h \subset \mathcal{U}$ of the test space $\mathcal{V}$ and the trial space $\mathcal{U}$. That is, let $\beta_1, \dots, \beta_n \in \mathcal{V}$ be a basis for $\mathcal{V}_h$ and $\alpha_1, \dots, \alpha_n \in \mathcal{U}$ be a basis for $\mathcal{U}$. Then we can write $v_h(\boldsymbol{x}) = \sum_{i=1}^n v_h^i \beta_i(\boldsymbol{x})$ and $u_h(\boldsymbol{x}) = \sum_{i=1}^n u_h^i \alpha_i(\boldsymbol{x})$. This reduces the problem to a finite-dimensional system of equations which we can solve for the vectors of coefficients $\boldsymbol{u}_h, \boldsymbol{v}_h \in \mathbb{R}^n$.

3.1.4 FEM Formulation of Harmonic Inpainting

Consider the harmonic inpainting problem on a domain $\Omega \subseteq \mathbb{R}^d$

$$\begin{aligned} -\Delta u(\boldsymbol{x}) &= 0, & \boldsymbol{x} &\in \Omega \setminus \mathcal{X}, \\ \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, & \boldsymbol{x} &\in \partial\Omega, \\ u(\boldsymbol{x}) &= f(\boldsymbol{x}), & \boldsymbol{x} &\in \mathcal{X}, \end{aligned} \tag{3.1}$$

where $|\mathcal{X}| = m$. The finite element method first multiplies the equation on both sides by a test function $v : \Omega \to \mathbb{R}$ and then integrates both sides over the problem domain Ω

$$-\int_\Omega v\Delta u \,\mathrm{dvol}^d = 0. \tag{3.2}$$

While one could directly apply a discretisation at this point, typically we first bring the equation to a weak form in order to reduce the highest order derivative. We can achieve this in the above by using the product rule

$$\nabla \cdot (v\boldsymbol{r}) = \nabla v \cdot \boldsymbol{r} + v\nabla \cdot \boldsymbol{r} \overset{\boldsymbol{r}=\nabla u}{\Longrightarrow} v\Delta u = \nabla \cdot (v\nabla u) - \nabla v \cdot \nabla u. \tag{3.3}$$

The next step is to integrate both sides and use the divergence theorem

$$\begin{aligned} -\int_\Omega v\Delta u \,\mathrm{dvol}^d &= \int_\Omega \nabla v \cdot \nabla u \,\mathrm{dvol}^d - \int_\Omega \nabla \cdot (v\nabla u) \,\mathrm{dvol}^d \\ &= \int_\Omega \nabla v \cdot \nabla u \,\mathrm{dvol}^d - \int_{\partial\Omega} v\partial_{\boldsymbol{n}} u \,\mathrm{d}\mathcal{H}^{d-1}. \end{aligned} \tag{3.4}$$

Thus, we have decreased the order of the spatial derivatives by half. Since $\partial_{\boldsymbol{n}} u = 0$ on $\partial\Omega$ (i.e. reflecting BCs) the boundary integral becomes zero. Then we end up with *the weak formulation* where the spatial derivatives are halved:

$$\forall v \in \mathcal{V}, \int_\Omega \nabla v \cdot \nabla u \,\mathrm{dvol}^d = 0, \quad u|_{\mathcal{X}} = v|_{\mathcal{X}}. \tag{3.5}$$

We can now replace u and v by their finite-dimensional approximations $u_h(\boldsymbol{x}) = \sum_{j=1}^n u_h^j \alpha_j(\boldsymbol{x})$ and $v_h(\boldsymbol{x}) = \sum_{i=1}^n v_h^i \beta_i(\boldsymbol{x})$. Plugging those in the weak formulation and using linearity yields

$$\sum_{i=1}^n \sum_{j=1}^n v_h^i u_h^j \underbrace{\int_\Omega \nabla \beta_i \cdot \nabla \alpha_j \,\mathrm{dvol}^d}_{W_{ij}} = 0. \tag{3.6}$$

The matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ is known as the stiffness matrix and it is a weak discretised version of the negated Laplacian (up to rescaling by the mass matrix). For simplicity assume that the basis functions β_j and α_j are interpolating over a set of points $\mathcal{Y}$ where $\mathcal{X} \subseteq \mathcal{Y}$. That is, we have the Lagrange property

$$\beta_j(\boldsymbol{y}_i) = \delta_{ij} = \alpha_j(\boldsymbol{y}_i), \quad \boldsymbol{y}_i \in \mathcal{Y}. \tag{3.7}$$

Then we can enforce the interpolation constraint $u|_{\mathcal{X}} = f|_{\mathcal{X}}$ directly

$$u_h(\boldsymbol{x}_i) = \sum_{j=1}^n u^j \alpha_j(\boldsymbol{x}_i) = u^i = f^i = f(\boldsymbol{x}_i), \quad \boldsymbol{x}_i \in \mathcal{X}. \tag{3.8}$$

Moreover, we require that v_h vanishes on the interpolation points

$$v_h(\boldsymbol{x}_i) = \sum_{j=1}^n v^j \beta_j(\boldsymbol{x}_i) = v^i = 0, \quad \boldsymbol{x}_i \in \mathcal{X}. \tag{3.9}$$

Denote by $\mathcal{I}$ the set $\{1,\ldots,n\}$, and let $\mathcal{C} \subseteq \mathcal{I}$ be the mask index set, i.e.,

$$i \in \mathcal{C} \implies \boldsymbol{y}_i \in \mathcal{X}.$$

We can now incorporate the interpolation constraints in (3.6)

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}} W_{ij} v_h^i u_h^j = 0 \implies \sum_{i\in\mathcal{I}\smallsetminus\mathcal{C}}\ \sum_{j\in\mathcal{I}\smallsetminus C} W_{ij} v_h^i u_h^j + \sum_{i\in\mathcal{I}\smallsetminus\mathcal{C}}\ \sum_{j\in\mathcal{C}} W_{ij} v_h^i f^j = 0. \tag{3.10}$$

Since we want the above to hold for any $v_h \in \mathcal{V}_h$, we can consecutively set v_h to be $\boldsymbol{e}_i$ for $i \in \mathcal{I} \smallsetminus \mathcal{C}$, which results in the system

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{W})\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}, \tag{3.11}$$

where $c_i = 1$ if $i \in \mathcal{C}$ and $c_i = 0$ otherwise, and $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$. Note that the structure of the system is similar to the one from the FDM method (1.5), which is unsurprising. The main difference is that our negated Laplacian discretisation $\boldsymbol{W}$ arises from the finite element method.

The only question that remains is how to construct β_i and α_i. In the next subsection we take $\alpha_i = \beta_i$ and use the piecewise-linear hat functions corresponding to the $\mathbb{P}_1$ elements for the basis.

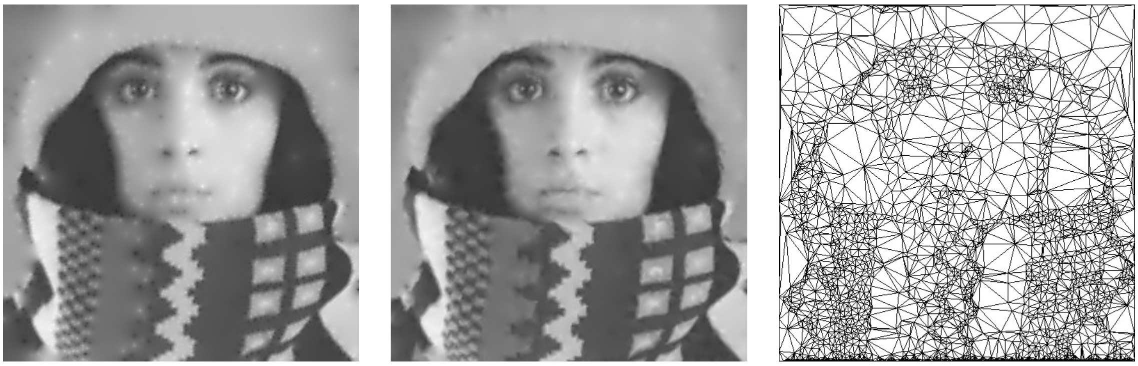
3.1.5 Implementation

Our FEM formulation is based on a triangular mesh; see Fig. 3.1, right. Each mask pixel is treated as a vertex in this mesh, and a subset, $\mathcal{Y}$, of the remaining non-mask pixels is selected as *unknown vertices*, based on the desired number of unknowns or runtime constraints. If all non-mask pixels were included as unknowns, the resulting discretisation would correspond to the standard 5-point stencil FDM formulation of the Laplace equation.

Once the vertex set $\mathcal{X} \cup \mathcal{Y}$ is defined, we construct a Delaunay triangulation [195]. The Delaunay property, which maximises the minimum angle in the triangulation, is beneficial for numerical stability, as it improves the conditioning of the system matrices used in both inpainting and tonal optimisation [208]. Additionally, this approach eliminates the need to store mesh connectivity explicitly, since the mesh can be reconstructed directly from the mask pixels $\mathcal{X}$ and unknown vertices $\mathcal{Y}$ [195].

We employ a *linear* FEM scheme: the reconstructed image is piecewise-linear within each triangle and continuous across edges and vertices. Grey or colour values are prescribed at mask pixels, while values at unknown vertices are computed by solving the linear system derived from the FEM formulation in (3.11). Typically, we select as many unknown vertices as there are mask pixels, i.e., $|\mathcal{Y}| = |\mathcal{X}|$, since our goal is to have $|\mathcal{X}| + |\mathcal{Y}| \ll N$, where $N = \mathit{width} \times \mathit{heigh}$, in order to get a highly efficient method.

Given a Delaunay mesh containing at least one mask pixel, the system matrix in (3.11) is symmetric and positive definite. Consequently, the solution can be efficiently



FDM inpainting FEM inpainting Delaunay triangulation

FIGURE 3.1: Harmonic inpainting with FDM and FEM. Note the logarithmic singularity suppression. Right: Delaunay triangulation used as the FEM mesh.

approximated using the conjugate gradient method [206]. The final image is obtained by linearly interpolating the solution within each triangle, in order to assign values to the remaining non-mask pixels that are not part of the mesh.

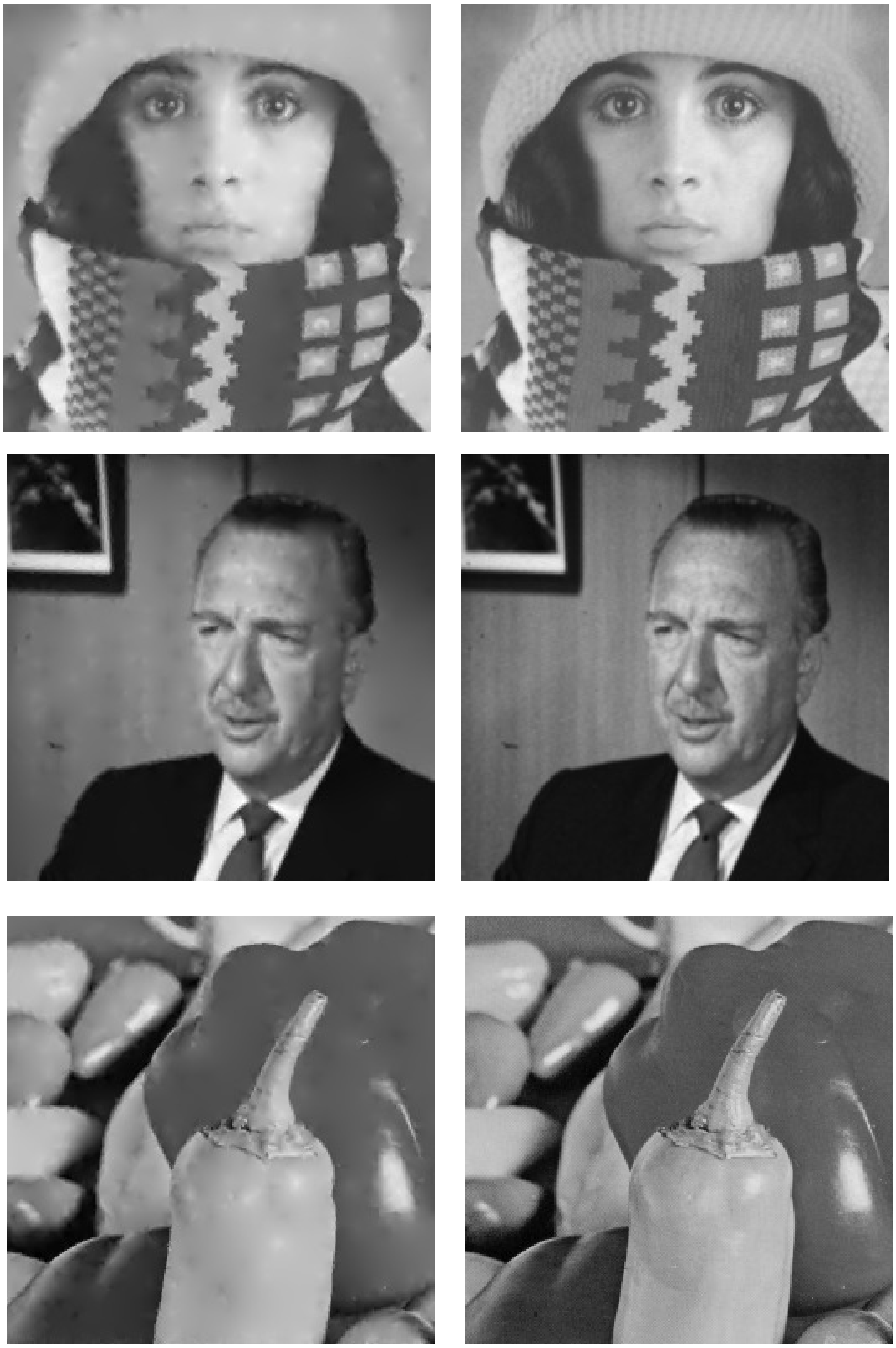
3.1.6 Experiments

In our experiments, we have tested mask densities of 4% and 10%, i.e., $|\mathcal{X}| = 4\% \cdot N$ and $|\mathcal{X}| = 10\% \cdot N$. As mentioned, we choose the FEM mesh such that $|\mathcal{Y}| = |\mathcal{X}|$, which results in a relatively well-conditioned system, as only half of the points are unknowns. In contrast, achieving such conditioning with FDM would require a mask density of 50%. Moreover, the resulting system is significantly smaller – approximately one-tenth to one-fifth of the size of the corresponding FDM system. This leads to an inpainting that is more than ten times faster for a 256×256 image compared to its FDM counterpart, with the performance gap increasing as image resolution grows (the better conditioning plays a significant role). The experiments w.r.t. performance were carried out on a system with a *Ryzen 4800H* CPU, however, the factor 10 doesn't generally change even on other hardware.

Representative inpainting results are shown in Fig. 3.1, Fig. 3.2, and Fig. 3.3. Notably, the logarithmic singularities characteristic of harmonic inpainting are absent in all cases – this is due to the regularising effect of choosing $|\mathcal{X}| + |\mathcal{Y}| \ll N$. Additional results on the acceleration of spatial and tonal optimisation with FEM inpainting are presented in Chapter 4.

3.1.7 Extensions

Since the publication of [58], we have explored several extensions, all of which have yielded negative results. Specifically, we have conducted experiments with the following approaches:



FEM original

Figure 3.2: FEM reconstructions with a 4% mask density and tonal optimisation. Note the lack of logarithmic singularities.

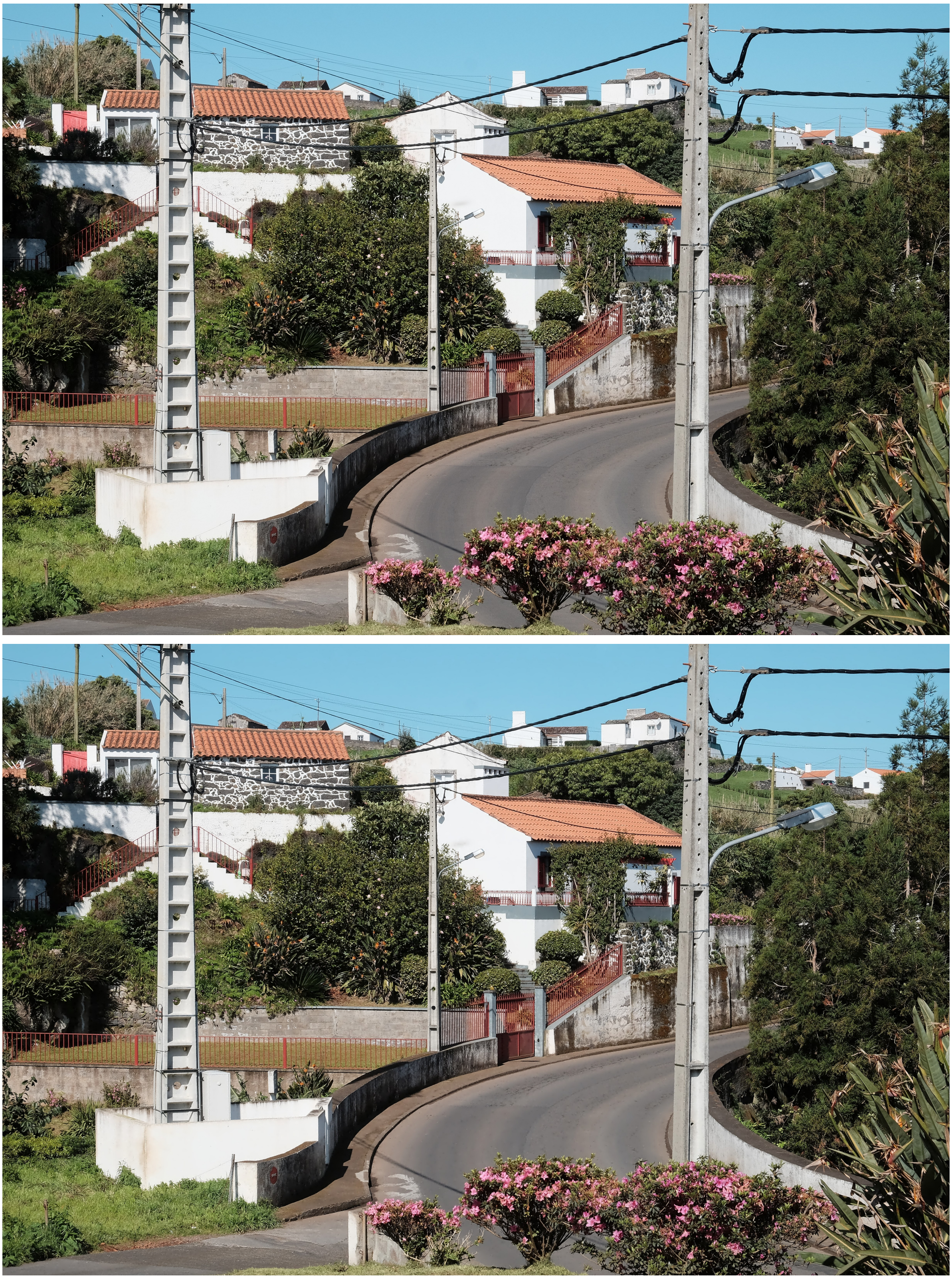
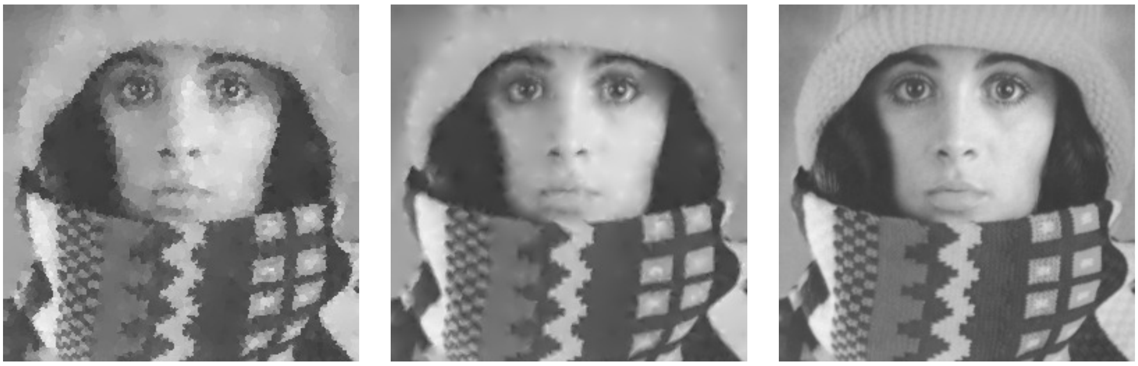


FIGURE 3.3: Top: A richly textured colour image of size 4896 × 3264 amounting to ca. 16 million pixels. Photo: J. Weickert. Bottom: Our FEM reconstruction with a 10% mask density.

- $\mathbb{P}_2$ triangle elements (quadratic polynomials) for harmonic inpainting,
- Morley, Bell, and Argyris elements for biharmonic inpainting,
- finite volume methods (FVM) and discontinuous Galerkin (DG) approaches.

Quadratic triangle elements $\mathbb{P}_2$ tend to increase the system's bandwidth, which leads to longer reconstruction times. In all of our tests, $\mathbb{P}_2$ elements do not perform better in terms of reconstruction quality compared to the $\mathbb{P}_1$ elements. Bell and Argyris elements provide C^1 continuity, enabling a conforming discretisation of the biharmonic inpainting problem. However, these did not yield better reconstruction results in terms of MSE and incurred significantly higher runtimes. This is due to the increased complexity: Bell elements have 18 degrees of freedom, and Argyris elements have 21, compared to just 3 for $\mathbb{P}_1$ elements.

Morley elements, piecewise-constant FVM, and discontinuous Galerkin methods produce discontinuous reconstructions (see Fig. 3.4), which negatively impact the MSE. Moreover, these elements do not offer any speed advantage over $\mathbb{P}_1$ elements. **These findings highlight that highly specialised schemes for specific PDEs do not necessarily translate into more efficient or higher-quality image reconstructions.**



FVM inpainted FEM inpainted original image

FIGURE 3.4: FVM vs FEM inpainting. Note the discontinuous reconstruction due to the piecewise-constant Voronoi elements. If we choose piecewise-linear triangle elements, the FVM inpainting is equivalent to the FEM one, since the two differ only by the mass matrix.

3.1.8 Polyharmonic Inpainting with $\mathbb{P}_1$ Elements

We briefly discuss our preferred discretisation for polyharmonic inpainting. Consider the biharmonic inpainting problem

$$\begin{aligned} (-\Delta)^2 u(\boldsymbol{x}) &= 0, & \boldsymbol{x} &\in \Omega \setminus \mathcal{X}, \\ \partial_{\boldsymbol{n}}(-\Delta) u(\boldsymbol{x}) &= 0, & \boldsymbol{x} &\in \partial\Omega, \\ \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, & \boldsymbol{x} &\in \partial\Omega, \\ u(\boldsymbol{x}) &= f(\boldsymbol{x}), & \boldsymbol{x} &\in \mathcal{X}. \end{aligned} \tag{3.12}$$

The weak formulation of the above problem involves a matrix

$$S_{ij} = \int_\Omega \Delta\beta_i \, \Delta\alpha_j \, \mathrm{dvol}^d. \tag{3.13}$$

That is, we need at least C^1 elements in general. However, if we set $v = -\Delta u$ we could decompose the problem into two coupled sub-problems (note that this is possible only for specific boundary conditions, such as the reflecting boundary conditions in our setting):

$$\begin{aligned} (-\Delta)v(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Omega \setminus \mathcal{X}, \\ \partial_{\boldsymbol{n}} v(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega, \\ (-\Delta)u(\boldsymbol{x}) &= v(\boldsymbol{x}), && \boldsymbol{x} \in \Omega, \\ \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega, \\ u(\boldsymbol{x}) &= f(\boldsymbol{x}), && \boldsymbol{x} \in \mathcal{X}. \end{aligned} \tag{3.14}$$

We can now discretise each of those separately, resulting in:

$$\boldsymbol{W}\boldsymbol{u} = \boldsymbol{M}\boldsymbol{v}, \quad (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{W}\boldsymbol{v} = \boldsymbol{0}, \quad \boldsymbol{C}\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}, \tag{3.15}$$

where $M_{ij} = \int_\Omega \beta_i \, \alpha_j \, \mathrm{dvol}^d$ is the mass matrix. Setting $\boldsymbol{v} = \boldsymbol{M}^{-1}\boldsymbol{W}\boldsymbol{u}$ and substituting in the second equation, while combining with the interpolating conditions yields

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{W}\boldsymbol{M}^{-1}\boldsymbol{W})\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}. \tag{3.16}$$

In other words $\boldsymbol{W}\boldsymbol{M}^{-1}\boldsymbol{W} \approx (-\Delta)^2$. We can generalise this to arbitrary integer powers of the Laplacian, leading to $\boldsymbol{W}(\boldsymbol{M}^{-1}\boldsymbol{W})^{q-1} \approx (-\Delta)^q$. To be more precise, the above should really be

$$(\boldsymbol{M}^{-1}\boldsymbol{W})^q \approx (-\Delta)^q. \tag{3.17}$$

However we are able to get rid of one $\boldsymbol{M}^{-1}$ due to the right-hand side having a zero in one of the equations, which yields a symmetric system matrix. This means that polyharmonic inpainting can be implemented as

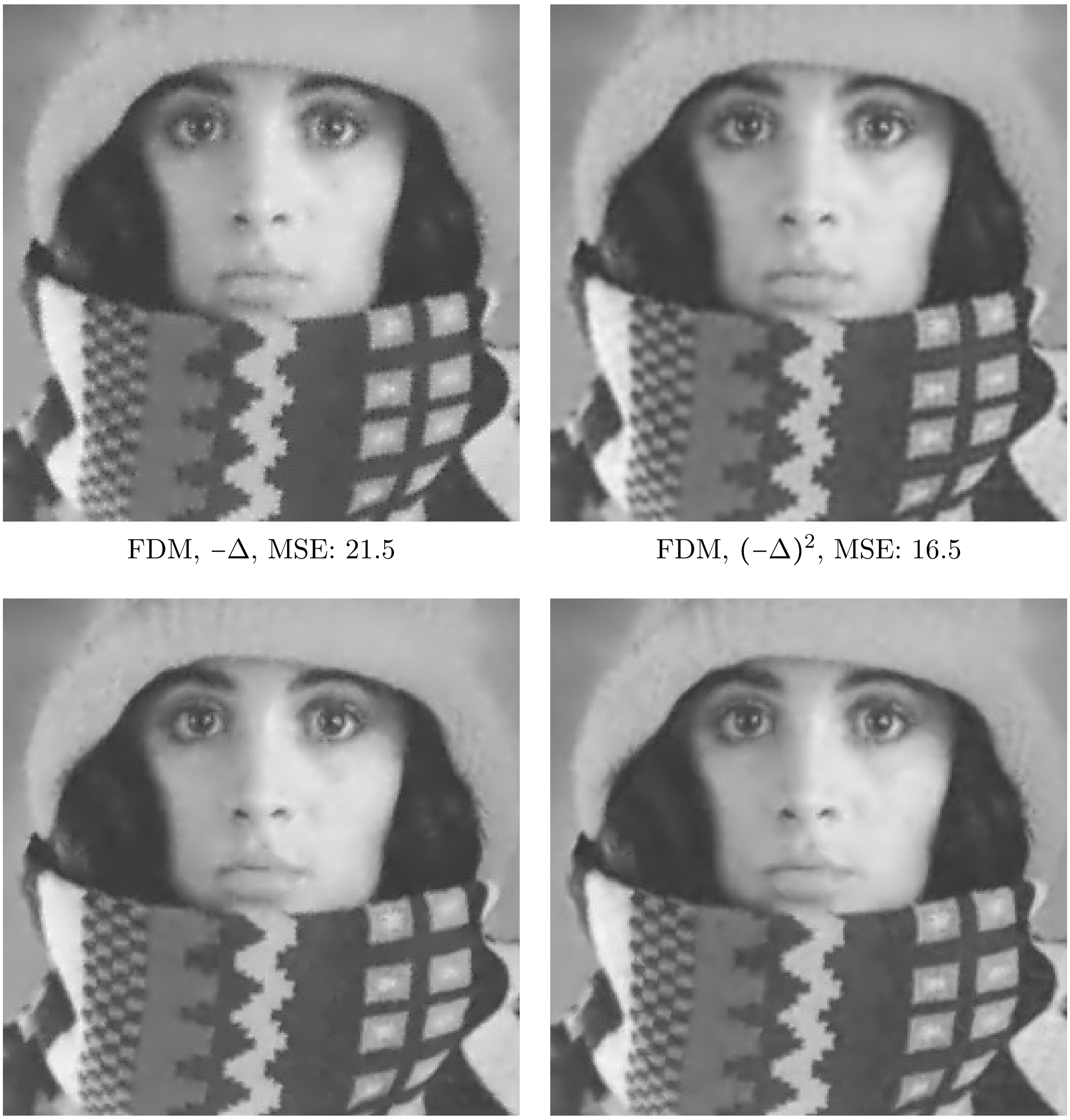
$$\left(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{W}(\boldsymbol{M}^{-1}\boldsymbol{W})^{q-1}\right)\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}. \tag{3.18}$$

For non-integer powers $\alpha \in \mathbb{R}$ we can use

$$\left(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})(\boldsymbol{M}^{-1}\boldsymbol{W})^{\alpha}\right)\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}. \tag{3.19}$$

We note that this discretisation is non-conforming since $\mathbb{P}_1$ elements do not form a subspace of the Sobolev space $H^q(\Omega)$ except when $q = 1$. Nevertheless, it is simple to implement, efficient, and results in reconstructions that are on par or better than ones using more complicated elements for biharmonic inpainting.

We illustrate some of the tonally optimised results with biharmonic inpainting in Fig. 3.5. Note that while FEM biharmonic inpainting achieves the same quality as FDM biharmonic inpainting, it is actually not better than FEM harmonic inpainting.



FDM, $-\Delta$, MSE: 21.5

FDM, $(-\Delta)^2$, MSE: 16.5

FEM, $-\Delta$, MSE: 16.3

FEM, $(-\Delta)^2$, MSE: 16.5

Figure 3.5: Harmonic and biharmonic inpainting with FDM and FEM at 5% (both are tonally optimised).

3.1.9 Conclusion

Our FEM harmonic inpainting implementation allows for notable efficiency gains over the standard FDM approach. Even for small images we can achieve a computational runtime improvement of over 10 times, and this gap grows larger as the resolution increases. Moreover, the regularisation from the FEM mesh strongly suppresses the logarithmic singularities associated with harmonic inpainting in 2D. On the other hand, biharmonic FEM or more complex elements do not provide any qualitative or runtime benefits. Natural avenues for future research are the evaluation of FEM for EED inpainting, as well as extensions to surface reconstruction similar to the work of Bae and Weickert [19].

3.2 Multigrid for Harmonic Inpainting

In [127, 128] we leverage domain decomposition and multigrid ideas for real-time inpainting of 4K images on the GPU. This addresses one of the main challenges related to the practicality of inpainting methods. In the current section we reproduce the part on multigrid from [127], as the main contributions of the author of the current thesis were specifically in that direction.

All of the contributions in [127, 128] regarding domain decomposition are due to the first author – Niklas Kämper. As such we provide only a short overview of the key takeaways regarding domain decomposition methods for inpainting in Section 3.2.3. For a complete treatment of the domain decomposition method we refer the reader to our works [127, 128].

3.2.1 Outline

We review the work on fast inpainting in the FDM setting in Section 3.2.2. We then provide a brief discussion of domain decomposition methods in Section 3.2.3. Basic notions of multigrid are presented in Section 3.2.4 and Section 3.2.5. Since we are working with a sparse mask, we give additional details on the construction of the coarser problems in Section 3.2.6 and Section 3.2.6.1. We describe a naive mask values downsampling approach in Section 3.2.6.2, however, the latter fails to preserve key properties of harmonic inpainting. In Section 3.2.6.3 we design a downsampling strategy that is specifically tailored to harmonic inpainting and allows us to achieve much better results. Experiments that study the runtime-to-density scaling and the runtime-to-resolution scaling are presented in Section 3.2.7. We conclude our discussion in Section 3.2.8, where we also discuss future work.

3.2.2 Related Work

In the following we discuss prior work on accelerating diffusion-based inpainting. In contrast to previous methods, we exploit the highly parallel nature of current GPUs and do not rely on any sort of temporal coherence of subsequent video frames. We integrate multigrid ideas, but complement them with a highly parallel domain decomposition method.

Domain Decomposition in Image Processing. While domain decomposition methods have not been used for sparse image inpainting, there exist a few domain decomposition approaches for image processing tasks, such as denoising [55, 136, 250], optic flow computation [137] or de-blurring [251].

Multigrid Methods. Multigrid methods [38, 40, 98, 228, 246] belong to the most efficient solvers for linear and nonlinear systems. They solve problems on multiple resolutions simultaneously, in order to achieve uniform convergence for both high- and low-frequency components of the error. Köstler et al. [139], Mainberger et al. [155], and Di Martino and Facciolo [71], use them for homogeneous diffusion inpainting on mildly parallel architectures such as multicore CPUs [155] or the *Playstation 3* [139]. For the smoothing operations on each level of the multigrid method, the listed works use a simple Gauss-Seidel iteration [206]. Köstler et al. [139] also considered nonlinear anisotropic diffusion with a coarse-to-fine multilevel method. While our approach also employs a (full) multigrid method, we improve the downsampling by adapting it to the inpainting operator, and we use a more sophisticated smoother, i.e. domain decomposition, that better utilises GPU parallelism.

Green's Functions. Another approach for accelerated inpainting goes back to Hoffmann et al. [113]. It is based on discrete Green's functions [16, 29, 60]. They can be used to decompose the inpainting problem into pixel-wise contributions from which the reconstruction can be assembled as a linear combination. This leads to an alternative system matrix of size $|K| \times |K|$ where $|K|$ is the number of known pixels, which determines the coefficients in the linear combination. This approach has the advantage that its runtime depends only on the number of mask pixels instead of the overall number of pixels. For very sparse inpainting data it can, thus, outperform multigrid methods. However, its system matrix is dense, and its size grows quadratically with the number of mask pixels. Thus, for 4K images and typical mask densities between 1% and 5% this approach is infeasible due to memory and runtime constraints. Kalmoun and Nasser's [125] approach is based on continuous Green's functions used in the method of fundamental solutions [132, 141]. They use a GMRES [206] solver with a fast multipole method [95]. The closed-form of the Green's functions coupled with the fast multipole method allow them to avoid storing the system matrix circumventing the memory constraints. Nevertheless, the inpaintings are still not real-time even at low resolutions such as 256×256.

Video Coding. In the context of video compression, Peter et al. [190] proposed a method for nonlinear anisotropic diffusion inpainting. It relies on fast explicit finite difference schemes [242], which are well-suited for parallelisation and benefit strongly from a good initialisation available from the previous frame. With this advantage, they were able to achieve real-time decoding of 640×480 videos on an *Nvidia GeForce GTX 460* GPU. In our work we do not utilise such temporal coherence which allows us to be more general. However, this also suggests that our methods can be made even faster when applied to videos.

Two other real-time capable video codecs that exploit temporal coherence go back to Andris et al. [13, 14]. They combine global homogeneous diffusion inpainting of keyframes with optic flow based prediction for the frames in between. In [14] they achieve real-time performance for FullHD colour videos on an *Intel Xeon CPU W3565@3.20GHz*, by solving the inpainting with a multilevel conjugate gradient method [36].

3.2.3 Domain Decomposition

Domain decomposition methods [73, 227] have similar convergence to block Jacobi (if additive Schwarz is used) or block Gauss-Seidel (if multiplicative Schwarz is used). Seeing as our method of choice in [127, 128] is optimised restricted additive Schwarz (ORAS), the best convergence rate that it can achieve is $O(\sqrt{\kappa})$, and in the worst case it is $O(\kappa)$, where κ is the condition number of the system matrix. As such additive Schwarz has slower convergence than conjugate gradients (CG) [206], and ORAS has the potential to achieve similar convergence to CG. Yet in our harmonic inpainting implementation, domain decomposition methods outperform conjugate gradients. The main reasons for this are cache locality and parallelisation.

The main bottleneck when applying iterative solvers such as CG to large sparse problems is memory reads and writes, as memory is not reused, and thus there are little benefits from cache (or local/shared memory on the GPU). Methods that have localised iterations such as additive and multiplicative Schwarz allow us to mitigate this, since we are able to perform multiple iterations on smaller blocks that can fit in cache (or local/shared memory on the GPU). This is true also for polynomial and block preconditioners.

The other benefit of additive Schwarz is that blocks are more or less independent of each other and we can solve the sub-problems in parallel. This is especially useful on GPUs where we have a large number of cores.

On the other hand, (single-level) domain decomposition methods cannot elide the fact that lower-frequency modes of the error are slow to decay. The latter is most naturally handled by multigrid methods, which allow a similar rate of error decrease over all frequencies. Thus next we focus on the implementation of a multigrid method for harmonic inpainting.

3.2.4 Two-Grid Cycle

Iterative solvers such as Jacobi [206], Gauss-Seidel [206], or the ORAS method [220] reduce the high frequency modes of the error efficiently, but they are much slower at decreasing the low frequencies. Multigrid addresses this by transferring the problem to a coarser grid where the low frequencies appear as higher frequencies which can be reduced more efficiently by the very same solvers. We first study the two grid formulation of multigrid which consist of iterating what's known as a *two-grid cycle*. Our two grids are the original fine grid with grid spacing h, and a coarser grid with spacing $H > h$. In our implementation, we set $H = 2h$.

We start the cycle $k + 1$ with an an approximation $\boldsymbol{u}_h^k$ of the solution $\boldsymbol{u}_h$ of the problem $\boldsymbol{A}_h\boldsymbol{u}_h = \boldsymbol{b}_h$ from the previous cycle k (or an initial approximation $\boldsymbol{u}_h^0$ for $k = 0$). Here $\boldsymbol{A}_h$ is the original system matrix and $\boldsymbol{b}_h$ is the right-hand side (both on the original fine grid). If we knew the true error $\boldsymbol{u}_h - \boldsymbol{u}_h^k$ we could correct our approximation to get the true solution: $\boldsymbol{u}_h^k + (\boldsymbol{u}_h - \boldsymbol{u}_h^k) = \boldsymbol{u}_h$. Consequently, our goal is to find an approximation of the error efficiently, by using the aforementioned idea of decimating different frequencies of the error over different grids. The error $\boldsymbol{u}_h - \boldsymbol{u}_h^k$ typically contains both high- and low-frequency components. The high frequencies can be reduced by performing a few iterations ϑ_{pre} of an iterative solver that dampens the high frequencies efficiently, also called a *smoother*. Typical choices are damped Jacobi or Gauss-Seidel, or ORAS. This results in the intermediate approximation of the solution

$$\boldsymbol{u}_h^{k+1/3} = \text{pre-smooth}(\boldsymbol{A}_h, \boldsymbol{b}_h, \boldsymbol{u}_h^k, \vartheta_{\text{pre}}). \tag{3.20}$$

Provided we have used enough iterations ϑ_{pre}, the error $\boldsymbol{e}_h^k = \boldsymbol{b}_h - \boldsymbol{u}_h^{k+1/3}$ will have only negligible high-frequency components. We can then represent it almost perfectly on the coarser grid. The transfer between the grids can be formalised using a restriction matrix $\boldsymbol{R}_h^H$ and a prolongation matrix $\boldsymbol{P}_H^h$. The two are designed so that sufficiently low frequency components are reproduced exactly, while higher frequency components are smoothed out to avoid aliasing.

A problem of the above formulation is that we do not know the error in practice, so we cannot transfer it to the coarser grid. However, we know the residual and its relation to the error:

$$\boldsymbol{r}_h^k = \boldsymbol{b}_h - \boldsymbol{A}_h \boldsymbol{u}_h^{k+1/3} = \boldsymbol{A}_h \boldsymbol{u}_h - \boldsymbol{A}_h \boldsymbol{u}_h^{k+1/3} = \boldsymbol{A}_h \left(\boldsymbol{u}_h - \boldsymbol{u}_h^{k+1/3}\right) = \boldsymbol{A}_h \boldsymbol{e}_h^k. \tag{3.21}$$

On the coarse grid this reads $\boldsymbol{A}_H \boldsymbol{e}_H^k = \boldsymbol{r}_H^k$, where $\boldsymbol{A}_H$ is an analogue of $\boldsymbol{A}_h$ on the lower level. While the coarse matrix $\boldsymbol{A}_H$ is usually constructed via the prolongation and restriction operators, this would be impractical in our case, as we usually do not explicitly store the system matrix. Instead, we directly discretise the inpainting problem on the coarser grid (see Section 3.2.6). We get the coarse residual $\boldsymbol{r}_H^k$ by transferring the fine grid residual $\boldsymbol{r}_h^k$ to the coarse grid:

$$\boldsymbol{r}_H^k = \boldsymbol{R}_h^H \boldsymbol{r}_h^k. \tag{3.22}$$

In the two-grid cycle we then solve the coarse grid problem for the error exactly:

$$\boldsymbol{A}_H \boldsymbol{e}_H^k = \boldsymbol{r}_H^k. \tag{3.23}$$

We note that the matrix $\boldsymbol{A}_H$ is twice smaller in each dimension compared to $\boldsymbol{A}_h$, thus the computational cost is also reduced. The coarse grid error $\boldsymbol{e}_H^k$ can then be transferred to the fine grid using the prolongation matrix, to get an approximation of the fine grid error: $\tilde{\boldsymbol{e}}_h^k = \boldsymbol{P}_H^h \boldsymbol{e}_H^k$. With this error approximation we can then correct our approximation of the solution to obtain

$$\boldsymbol{u}_h^{k+2/3} = \boldsymbol{u}^{k+1/3} + \tilde{\boldsymbol{e}}_h^k = \boldsymbol{u}^{k+1/3} + \boldsymbol{P}_H^h \boldsymbol{e}_H^k. \tag{3.24}$$

As a last step, a post-smoothing with ϑ_{post} iterations is applied to smooth any high frequency error components that have potentially been introduced by the previous correction step:

$$\boldsymbol{u}_h^{k+1} = \text{post-smooth}(\boldsymbol{A}_h, \boldsymbol{b}_h, \boldsymbol{u}_h^{k+2/3}, \vartheta_{\text{post}}). \tag{3.25}$$

All of these steps together result in Algorithm 1. While solving $\boldsymbol{A}_H \boldsymbol{x}_H = \boldsymbol{b}_H$ is generally computationally cheaper than solving $\boldsymbol{A}_h \boldsymbol{x}_h = \boldsymbol{b}_h$, for high resolution images even the coarse problem is expensive. Thus, one often considers recursively applying this process

Algorithm 1. Two-Grid Cycle

```
Input : current approximation $\boldsymbol{u}_h^k$
Output : improved approximation $\boldsymbol{u}_h^{k+1}$

1. Pre-smooth: $\boldsymbol{u}_h^{k+1/3} = \text{smoother}(\boldsymbol{A}_h, \boldsymbol{b}_h, \boldsymbol{u}_h^k, \vartheta_{\text{pre}})$
2. Compute Residual: $\boldsymbol{r}_h^k = \boldsymbol{b}_h - \boldsymbol{A}_h \, \boldsymbol{u}_h^{k+1/3}$
3. Restriction: $\boldsymbol{r}_H^k = \boldsymbol{R}_h^H \, \boldsymbol{r}_h^k$
4. Coarse Solve: $\boldsymbol{A}_H \, \boldsymbol{e}_H^k = \boldsymbol{r}_H^k$
5. Prolongation + Correction: $\boldsymbol{u}_h^{k+2/3} = \boldsymbol{u}_h^{k+1/3} + \boldsymbol{P}_H^h \, \boldsymbol{e}_H^k$
6. Post-smooth: $\boldsymbol{u}_h^{k+1} = \text{smoother}(\boldsymbol{A}_h, \boldsymbol{b}_h, \boldsymbol{u}_h^{k+2/3}, \vartheta_{\text{post}})$
```

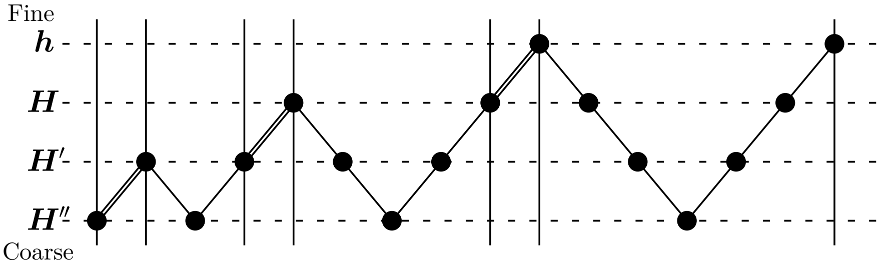


FIGURE 3.6: Full multigrid scheme. Example with four resolution layers and a single V-cycle for each level. The doubled lines represent the FMG prolongations to initialise the V-cycle for the next finer level. Multigrid schematic courtesy of Niklas Kämper.

over a sequence of grids with decreasing resolution until a grid of a sufficiently low resolution is reached. The latter results in an almost optimal workload of $O(N \log N)$ for computing the solution. Another issue is how to choose a good initial guess $\boldsymbol{u}_h^0$. It turns out that a good quality guess can be constructed efficiently by starting from the coarsest level. The recursive approach with this initial guess has an optimal workload of $O(N)$. Both of these ideas are discussed in the following subsection.

3.2.5 V-Cycles and Full Multigrid

Instead of using only two resolution levels, we can extend the two-grid cycle to multiple levels, by replacing the solve on the coarse level with another correction step on an even coarser level. By iterating this process recursively until we reach a level that is coarse enough for the lowest possible frequencies in our problem, we obtain a *V-cycle*. This allows us to efficiently dampen different frequencies of the error at different resolutions.

We can improve the multigrid scheme even further, by providing a good but inexpensive initialisation for the solution on the fine resolution level. Such an initialisation can be obtained by computing an inpainting solution on a coarser grid and prolongating it to the fine resolution. This can be extended to a complete coarse-to-fine initialisation strategy. Starting from a very coarse resolution grid, we successively refine the problem, where inpainting solutions from coarser levels are interpolated and used as an initialisation for finer ones. At each level a V-cycle is used to solve the resulting linear system. This coarse-to-fine method with the error correction steps is called *full multigrid (FMG)*. Fig. 3.6 shows an illustration how the different grids are traversed in such an FMG scheme. However, restriction and prolongation between different grids can be costly in terms of runtime on parallel hardware such as GPUs, as they consist mainly of global memory operations. To reduce the amount of restriction and prolongation operations without losing the advantages of FMG, we propose to skip the V-cycles on the lower levels and just use a single pre-smoothing iteration instead. This reduced FMG scheme, is depicted in Fig. 3.7. For homogeneous diffusion inpainting, this reduced FMG scheme is sufficient for a good convergence and improves the overall runtime compared to the non-reduced scheme.

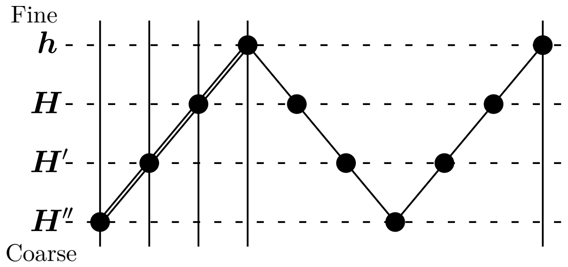


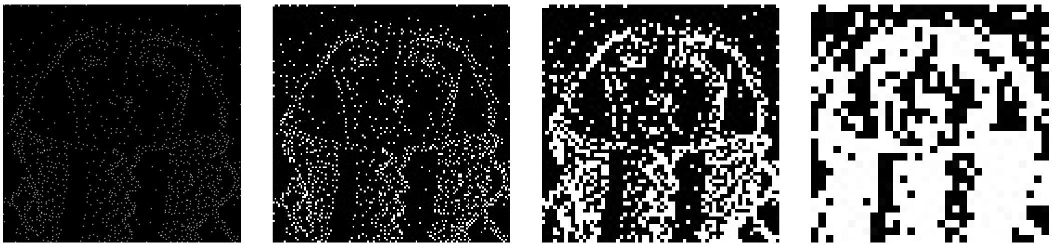
FIGURE 3.7: Reduced full multigrid scheme. The initial guess is constructed in a coarse-to-fine manner, also known as one-way or cascadic multigrid [36]. Then we continue with additional V-cycle correction steps (a single V-cycle is visualised above). Multigrid schematic courtesy of Niklas Kämper.

3.2.6 Lower Resolution Problems Construction

Full multigrid requires constructing a problem $\boldsymbol{A}_H\boldsymbol{u}_H = \boldsymbol{b}_H$ on the coarser grid that is similar in structure to $\boldsymbol{A}_h\boldsymbol{u}_h = \boldsymbol{b}_h$ on the finer grid. To achieve this, we employ a downsampling ratio of two in each spatial direction, ensuring that the discrepancy between the two problems remains minimal.

In the multigrid literature, the standard assumption is that the computational domain has a rectangular boundary. Under this assumption, the geometry and connectivity of the boundary are preserved during downsampling. In our setting, however, only the reflecting Neumann boundary coincides with the rectangular image boundary. The Dirichlet boundary, defined by the mask $\boldsymbol{c}$, can be arbitrarily complex and typically consists of scattered pixels with highly non-uniform density across the image domain. Consequently, both the geometry and the connectivity of the Dirichlet boundary change significantly when the grid is coarsened.

This distinction is important, as it is well known that grid transfer operations in multigrid methods must be carefully adapted near boundaries to achieve optimal performance [246].



256 × 256 128 × 128 64 × 64 32 × 32

FIGURE 3.8: Mask restriction example on *trui* with a 2% mask density. With each resolution level the mask density increases while the connectivity of the mask changes.

3.2.6.1 Mask Downsampling

In order to construct coarser versions of the mask $\boldsymbol{c}$, we use a specific dyadic downsampling strategy: a pixel on the coarser level is marked as a mask pixel if *any* of its corresponding finer-level pixels are mask pixels. This approach resembles max pooling rather than average pooling. For an illustration of this process, see Fig. 3.8.

This strategy is conceptually similar to the method used in [164], although their work involves simpler boundary configurations. We observe that downsampling in this manner increases the density of the boundary set, which generally improves the efficiency of solving the coarse-grid problem. However, this also alters the geometry and even connectivity of the boundary on coarser levels, resulting in a coarse discretisation that deviates significantly from the original fine-grid problem.

This discrepancy helps explain why a multigrid-preconditioned conjugate gradient solver is not well suited to our problem, and why aggressive downsampling ratios – commonly used in standard multilevel domain decomposition methods – are suboptimal in our setting.

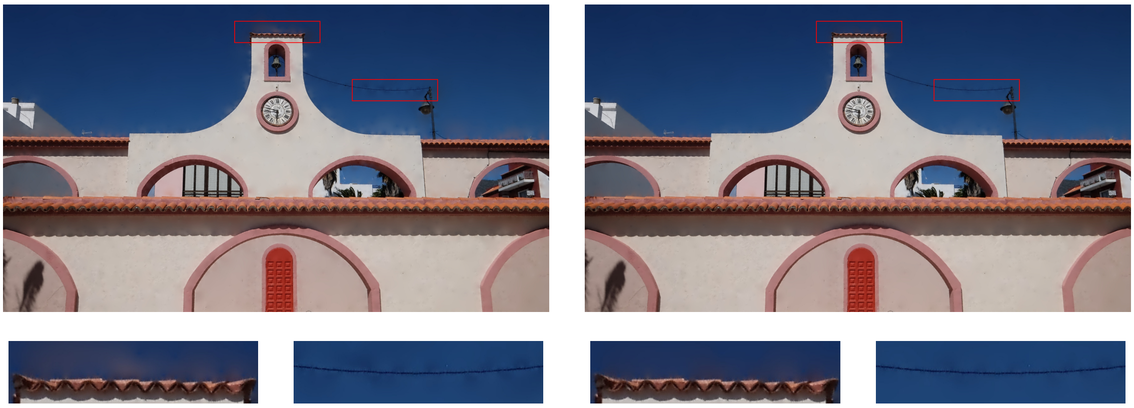
3.2.6.2 Naïve Mask Values Downsampling

Besides constructing the mask itself, we also need to determine lower-resolution counterparts $\boldsymbol{b}_H = \boldsymbol{C}_H \boldsymbol{f}_H$ for the mask values $\boldsymbol{b}_h = \boldsymbol{C}_h \boldsymbol{f}_h$.

A naïve downsampling approach averages the values of all mask pixels within each 2 × 2 neighbourhood on the finer level. If a neighbourhood contains no mask pixels, the corresponding coarse-level pixel is not considered part of the mask, and its right-hand side value is set to zero.

As an example, the naïve coarse-level right-hand side value $b_H^{2,2}$ for the coarse pixel $c_H^{2,2}$ is computed by averaging the values of the corresponding finer-level mask pixels in its 2 × 2 region.

$$\begin{array}{|c|c|} \hline c_h^{3,3} & c_h^{4,3} \\ \hline c_h^{3,4} & c_h^{4,4} \\ \hline \end{array}\,, \quad \begin{array}{|c|c|} \hline b_h^{3,3} & b_h^{4,3} \\ \hline b_h^{3,4} & b_h^{4,4} \\ \hline \end{array}\,, \quad b_H^{2,2} = \frac{c_h^{3,3} b_h^{3,3} + c_h^{3,4} b_h^{3,4} + c_h^{4,3} b_h^{4,3} + c_h^{4,4} b_h^{4,4}}{\max\left(1, c_h^{3,3} + c_h^{4,3} + c_h^{3,4} + c_h^{4,4}\right)}. \tag{3.26}$$



(a) naïve downsampling (b) modified downsampling

Figure 3.9: Illustration of the leakage caused by naïve dyadic downsampling for a 2% mask using only one-way multigrid (i.e. nested iteration).

As one might expect with such a complex boundary, the simple averaging and downscaling approach described above is not ideal. A naïve average combines up to four mask values from the finer level to compute the value of a single mask pixel on the coarser level. However, a key observation specific to homogeneous diffusion inpainting is that the influence of a mask pixel is localised to regions enclosed by other mask pixels. This implies that naïve downsampling can cause mask pixels with originally localised effects on the fine grid to exert influence beyond their intended boundaries on the coarse grid. This phenomenon manifests as noticeable leakage across boundaries, leading to pronounced colour bleeding near edges, as illustrated in Fig. 3.9. To rectify this, we modify the downsampling procedure for the mask values by allowing mask pixels to suppress the contribution of their neighbours.

3.2.6.3 Modified Mask Values Downsampling

If the four neighbours of a mask pixel are also mask pixels, then its value should not be averaged on the coarser level, since its influence is already suppressed by its surroundings. Importantly, these neighbours can be either fine-scale or coarse-scale mask pixels, as long as they are direct neighbours, they contribute to suppressing the pixel's influence.

As an example, consider the fine-level mask pixel $c_h^{3,3}$. Its direct neighbours include the fine-grid pixels $c_h^{3,4}$ and $c_h^{4,3}$, as well as the coarse-grid pixels $c_H^{1,2}$ and $c_H^{2,1}$. An illustration of this configuration is shown below, where the coarse pixel $c_H^{2,2}$ is decomposed into its fine-scale constituents: $c_h^{3,3}$, $c_h^{3,4}$, $c_h^{4,3}$, and $c_h^{4,4}$.

$$\begin{array}{|c|c|c|c|}
\hline
* & \multicolumn{2}{c|}{c_H^{2,1}} & * \\
\hline
 & c_h^{3,3} & c_h^{4,3} & \\
\cline{2-3}
c_H^{1,2} & c_h^{3,4} & c_h^{4,4} & c_H^{3,2} \\
\hline
* & \multicolumn{2}{c|}{c_H^{2,3}} & * \\
\hline
\end{array} \tag{3.27}$$

To model this suppression effect, we assign each fine-grid pixel a weight in the averaging process based on the number of its non-mask pixel neighbours. This leads to a modified downsampling rule for computing the coarse-level right-hand side value $b_H^{2,2}$:

$$\begin{aligned} w_h^{3,3} &= c_h^{3,3}\left(4 - c_H^{1,2} - c_H^{2,1} - c_h^{4,3} - c_h^{3,4}\right), \quad & w_h^{3,4} &= c_h^{3,4}\left(4 - c_H^{1,2} - c_h^{3,3} - c_h^{4,3} - c_H^{2,3}\right), \\ w_h^{4,3} &= c_h^{4,3}\left(4 - c_h^{3,3} - c_H^{2,1} - c_H^{3,2} - c_h^{3,4}\right), \quad & w_h^{4,4} &= c_h^{4,4}\left(4 - c_h^{3,3} - c_h^{3,3} - c_H^{3,2} - c_H^{2,3}\right), \end{aligned} \tag{3.28}$$

$$b_H^{2,2} = \frac{w_h^{3,3} b_h^{3,3} + w_h^{3,4} b_h^{3,4} + w_h^{4,3} b_h^{4,3} + w_h^{4,4} b_h^{4,4}}{\max\left(1, w_h^{3,3} + w_h^{3,4} + w_h^{4,3} + w_h^{4,4}\right)}. \tag{3.29}$$

The effect of this modification is illustrated in Figure 3.9, which exhibits significantly less color bleeding than Figure 3.9. While the artifacts in the latter could be mitigated through additional V-cycles or more sophisticated smoothers, the modified downsampling approach reduces such issues at a much lower computational cost.

3.2.7 Experiments

After describing the technical details of our experimental setup in Section 3.2.7.1, we illustrate its performance scaling behaviour in terms of the mask density in Section 3.2.7.3, and in terms of the resolution in Section 3.2.7.4.

3.2.7.1 Experimental Setup

For the experimental evaluation of our full multigrid ORAS inpainting, we compare against the cascadic multigrid ORAS approach from [126] and CG, as these methods are suitable for parallelisation and can be efficiently implemented on a GPU. The experiments for ORAS plus multigrid were conducted on a machine with an *AMD Ryzen 5900X@3.7GHz* CPU and an *Nvidia GeForce GTX 1080 Ti* GPU. We used state-of-the-art optimised inpainting masks, obtained with a Delaunay densification strategy [128]. As optimised inpainting masks have local mask pixel densities of significant variance across the image domain, it is much harder to inpaint images with them, compared to regular or random masks. Thus, they are representative of the general case and we restrict our experiments to them. We achieve an average runtime of 14.3 milliseconds, which corresponds to **a frame rate of nearly 70 frames per second**.

3.2.7.2 Stopping Criterion

As a stopping criterion we use the relative decay of the Euclidean norm of the residual. We note that this alone is not a good indicator of the true reconstruction error, which is illustrated in Fig. 3.10. It shows the approximation quality in terms of the mean squared error (MSE) with respect to a fully converged inpainting result for a relative residual norm of 10^{-3}. For both single level methods, we get MSEs of more than 1, which shows that they are clearly not fully converged and would require a much stricter stopping

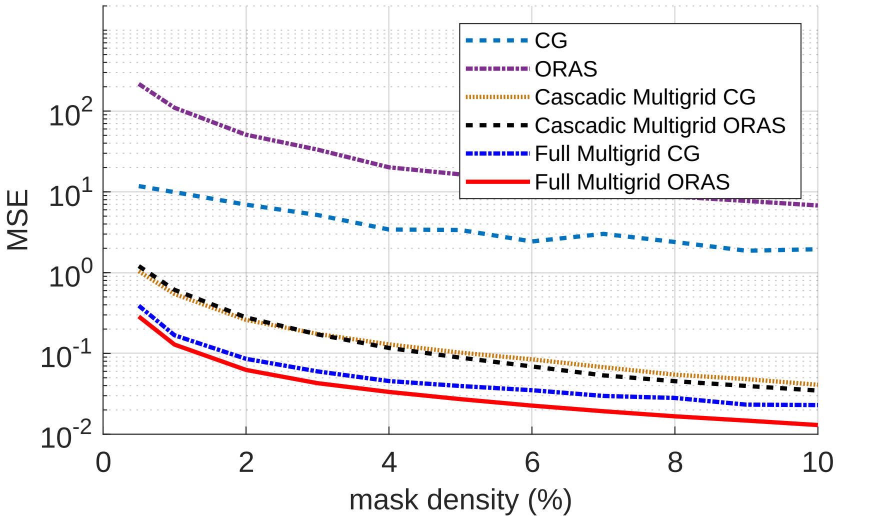


FIGURE 3.10: MSE between the inpainting approximation and a converged inpainting for a stopping criterion of 10^{-3}. The full multigrid methods are closest to the converged inpainting – with our full multigrid ORAS method performing the best. Both single level methods are clearly not fully converged and would require a much stricter stopping criterion for a reasonable approximation quality. Plot courtesy of Niklas Kämper.

criterion for a reasonable approximation quality. The multigrid methods, on the other hand, offer significantly lower approximation errors, for the same relative residual norm of 10^{-3}. This also shows that the full multigrid methods perform better than the cascadic ones. Due to this, we only consider multigrid methods from this point on, as they are clearly superior. We use the relative residual norm of 10^{-3} as the stopping criterion, as we have seen that this is sufficient for a reasonable approximation quality.

3.2.7.3 Mask Density Runtime Scaling

Fig. 3.11 shows the corresponding inpainting runtimes. We observe that the domain decomposition methods clearly outperform their corresponding CG-based solvers by more than a factor of 4. This demonstrates the capabilities of domain decomposition methods over simpler algorithms. While the performance of the cascadic and the full multigrid ORAS methods is similar for a mask density of 10%, our full multigrid method clearly outperforms the cascadic approach for lower mask densities. This is because even though both methods use a coarse-to-fine initial guess, only full multigrid performs V-cycles that efficiently reduce lower frequency error modes. Since for sparser masks the mask pixel contributions are lower frequency, those masks benefit much more from the full multigrid approach. This allows us to perform real-time inpainting with 30 frames per second for all tested densities and even 60 frames per second for mask densities higher than 2%.

3.2.7.4 Resolution Runtime Scaling

To evaluate our inpainting methods over different image sizes, we conduct experiments over resolutions ranging from 480 × 270 to 3840 × 2160 with 5% masks. The results are shown in Fig. 3.12. We can see that similar to the results from Fig. 3.11, our ORAS-based inpainting methods are over four times faster than their CG-based counterparts on all image resolutions. While both of our domain decomposition methods are able to inpaint 4K images in real-time, with at least 30 frames per second, the CG-based methods achieve this only for FullHD resolutions. Fig. 3.12 also reveals that, at least for higher image resolutions, all four methods show a nearly linear behaviour in the double logarithmic plot. As the slope is approximately 1 for all methods, we observe an ideal linear scaling behaviour.

3.2.8 Conclusion

One of the biggest challenges in inpainting-based compression has been the high computational complexity associated with inpainting algorithms. While qualitatively those are a viable alternative to transform-based approaches, standard solvers result in long inpainting runtimes for larger images. Our work substantially alleviates this issue. By adapting one of the most efficient concepts of numerical analysis – domain decomposition – and embedding it into a multigrid scheme, we are able to take advantage of the computing power of modern parallel hardware. This has resulted in the most efficient solver for sparse homogeneous diffusion inpainting. We have achieved, for the first time ever, homogeneous diffusion inpainting at 60 frames per second for 4K colour images on a contemporary GPU (in actuality four generations old). This suggests that even 8K resolution in real-time would be possible with the latest generation of GPUs. Furthermore, most state-of-the-art methods for data optimisation on the encoding side rely on multiple inpaintings. Thus, our work not only benefits the decoding side but is also able to improve the runtime of encoding methods. This demonstrates that inpainting-based compression has left its infancy to become a serious alternative to classical transform-based codecs not only in terms of quality but also for real-time applications.

Our experience with homogeneous diffusion inpainting suggests that approaches like ours, which judiciously adapt multiple numerical methods and exploit modern parallel architectures, could also be successfully transferred to numerous other image processing tasks.

Last but not least, while our work mainly focuses on mathematically sound algorithms, we have nevertheless extracted interesting insights about the structure of homogeneous diffusion inpainting and its interplay with numerical methods. We have seen that devising methods that attempt to preserve continuous properties such as the localisation of the Dirichlet boundary's influence on coarser levels pays off. We have also observed the relationship between mask density, cascadic, and full multigrid methods. The latter are able to achieve much lower runtimes for low densities, since the basis functions at low densities are inherently smoother. Finally, we noted that the widely used stopping

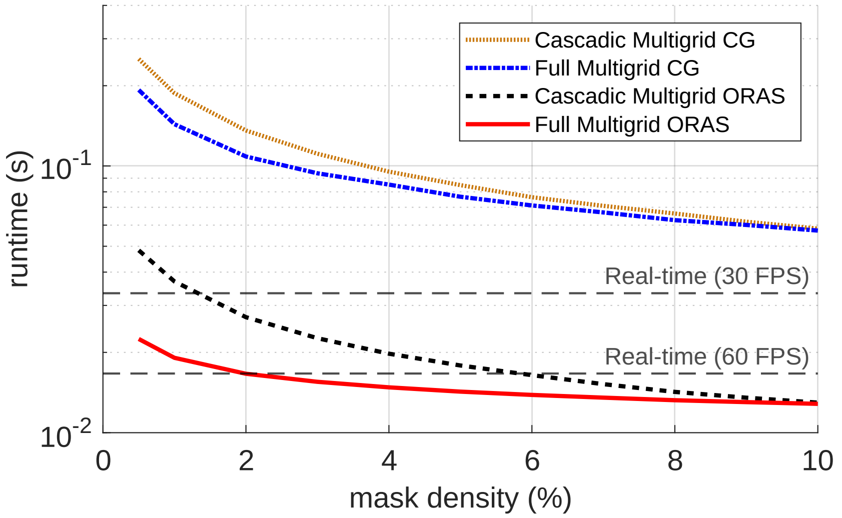


FIGURE 3.11: Runtimes of cascadic and full multigrid methods for a stopping criterion of 10^{-3}. Our ORAS-based methods are more than 4 times faster than their corresponding CG counterparts. Only our full multigrid ORAS method achieves real-time performance for all densities. Plot courtesy of Niklas Kämper.

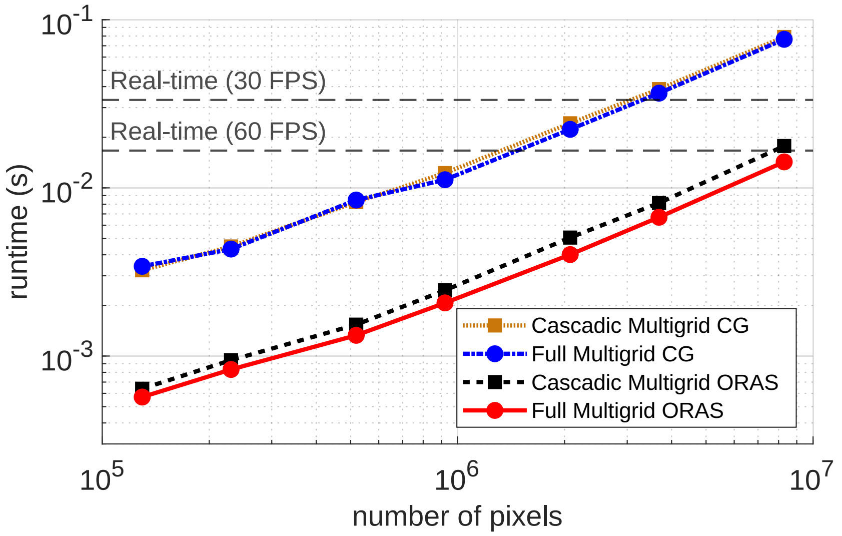


FIGURE 3.12: The horizontal dashed lines represent real-time inpainting with 30 and 60 frames per second. Our full multigrid ORAS method achieves more than 60 frames per second for a 4K resolution (last data point). With CG-based methods we achieve real-time performance only up to a FullHD resolution (third data point from the right) and only with approximately 45 frames per second. Plot courtesy of Niklas Kämper.

criterion based on the relative residual can be highly misleading, especially in the context of image processing while using different classes of solvers.

In our ongoing research, we are working on extending our domain decomposition approach to more sophisticated inpainting operators, such as anisotropic nonlinear diffusion. It has been shown that those can offer an improved reconstruction quality compared to the simple homogeneous diffusion inpainting.

Chapter 4

Data Optimisation Algorithms

As we emphasised in Chapter 1, data optimisation algorithms are crucial for the quality of the reconstruction of inpainting methods. The main problem that we aim to solve is

$$\mathcal{X}^* \in \operatorname*{arg\,min}_{|\mathcal{X}|=m} \|u(\mathcal{X}, \boldsymbol{g}^*(\mathcal{X})) - f\| \quad \text{such that} \quad \boldsymbol{g}^*(\mathcal{X}) \in \operatorname*{arg\,min}_{\boldsymbol{g}\in\mathbb{C}^m} \|u(\mathcal{X}, \boldsymbol{g}) - f\|, \tag{4.1}$$

where spatial optimisation (over $\mathcal{X}$) follows tonal optimisation (over $\boldsymbol{g}$). We often replace the latter by the simpler problem (4.2) where the spatial optimisation is done with respect to an interpolating reconstruction $u(\mathcal{X}, f|_{\mathcal{X}})$, followed by a single tonal optimisation step

$$\boldsymbol{g}^* \in \operatorname*{arg\,min}_{\boldsymbol{g}\in\mathbb{C}^m} \|u(\mathcal{X}^*, \boldsymbol{g}) - f\| \quad \text{such that} \quad \mathcal{X}^* \in \operatorname*{arg\,min}_{|\mathcal{X}|=m} \|u(\mathcal{X}, f|_{\mathcal{X}}) - f\|. \tag{4.2}$$

Despite of the substitution in the above, the goal remains to minimise

$$\min_{|\mathcal{X}|=m} \min_{\boldsymbol{g}\in\mathbb{C}^m} E(\mathcal{X}, \boldsymbol{g}), \quad E(\mathcal{X}, \boldsymbol{g}) = \|u(\mathcal{X}, \boldsymbol{g}) - f\|, \tag{4.3}$$

which is the energy being minimised by (4.1), while in the worst case (4.2) may yield a minimum only as good as the minimum of

$$\min_{|\mathcal{X}|=m} E(\mathcal{X}), \quad E(\mathcal{X}) = \|u(\mathcal{X}, f|_{\mathcal{X}}) - f\|. \tag{4.4}$$

Spatial Optimisation. Throughout the rest of the chapter we focus on the discrete setting where $\boldsymbol{f}, \boldsymbol{u}(\mathcal{X}, \cdot) \in \mathbb{R}^N$ are the values on the N vertices of some mesh – typically the values of the image pixels or the values of some function defined at the vertices of a triangular mesh. The search space for the spatial optimisation problem in that setting has size

$$\binom{N}{m} = \frac{N!}{m!(N-m)!} = O\left(\left(\frac{\mathrm{e}\cdot N}{m}\right)^m\right), \tag{4.5}$$

which precludes us from performing an exhaustive search even for small images (e.g. $N = 256 \cdot 256$) with a fairly low mask density (e.g. $m = 5\% \cdot N$). There are, of course, special cases where the above problem is easy to solve. For example, if we consider an orthogonal (unitary) transform, the best m-term approximation problem reduces to just sorting the absolute values of the N transformed coefficients and choosing the m

largest. Since the linear operators that we considered can be very far from orthogonal (the condition number of our matrices is very large even for harmonic inpainting), or are nonlinear, this is inapplicable in our setting. Consequently, we have to rely on some sort of greedy algorithms or other (meta-)heuristics in order to find an approximate solution of (4.1) or (4.2).

Tonal Optimisation. Generally, the tonal optimisation problem (i.e., the optimisation with respect to $\boldsymbol{g}$) is much simpler – for a linear inpainting (with linear features) and $\| \cdot \|$ being the 2-norm, this reduces to a linear least-squares problem. We can compute its solution by solving a linear system. In contrast, the spatial optimisation problem remains just as hard in the linear setting. Nevertheless, the main challenge in tonal optimisation is solving the normal equations efficiently, since even in the linear setting the system matrix is of size $m \times m$ and is typically dense – we cannot even store the latter in memory for moderately large images.

Publications. The contents of the current chapter are partially based on our data optimisation for FEM [58], the improvements of the latter in [128] for FDM, and the generalisation of the problem to feature inpainting from [120]. We further extend the approach to our generalisation of the feature inpainting framework proposed in Chapter 2. As such, the algorithms in the current chapter subsume our previous work and also support nonlinear features. There is one exception: We do not discuss the domain decomposition extension of the tonal optimisation, since the main contribution in the latter is due to the first author of [128] – Niklas Kämper. Instead we focus on our contributions related to the problem formulation, nested Krylov solvers, and the Richardson iteration for the construction of the initial guess. We also provide extensions that do not appear in [58, 120, 128].

4.1 Tonal Optimisation

Tonal optimisation is an important algorithmic step for achieving good quality reconstructions. Often we can improve the MSE by 30% for spatially well-optimised masks, and even larger improvements are to be had if the spatial optimisation did a poor job.

We mainly focus on tonal optimisation for linear inpainting with linear features, as this is the setting with which our publications on the topic [58, 120, 128] are concerned with. Our work on finite elements [58] describes a precursor to the tonal optimisation methods developed in [120, 128]. Nevertheless, we also briefly discuss the FEM formulation, as it differs in some aspects from the FDM approach, due to the interpolation involved in FEM.

4.1.1 Outline

We go over the related work on tonal optimisation for sparse inpainting in Section 4.1.2. This is followed by the formulation of the linear least squares problem for the linear feature inpainting framework in Section 4.1.3 – the latter is based on our work [120]. In Section 4.1.4 we discuss the application of Krylov methods to the "outer" tonal optimisation problem, while in Section 4.1.5 we discuss the formulation and solution of the "inner" inpainting-like problems. In Section 4.1.6 we describe a way to construct a good initial iterate for the tonal optimisation based on [128], and also extend this approach to the setting of feature inpainting. Experiments illustrating the performance of our algorithms in the FEM and FDM setting are presented in Section 4.1.7, Section 4.1.8, and Section 4.1.9. We conclude our discussion of tonal optimisation in Section 4.1.10, where we also mention new potential research directions.

4.1.2 Related Work

Here we review some of the related work on tonal optimisation for image inpainting. For brevity of notation we set $\boldsymbol{R} = \tilde{\boldsymbol{B}}^{-1}\tilde{\boldsymbol{P}}$ to be the reconstruction or inpainting matrix for a linear inpainting process.

Inpainting Echoes. For smaller images, the matrix $\boldsymbol{R}$ can be computed and stored explicitly. Its columns, known as inpainting echoes [156], are precomputed and stored by Mainberger et al. [156], effectively forming the matrix $\boldsymbol{R}$. Although they mention that solving the normal equations involving $\boldsymbol{R}^\top\boldsymbol{R}$ with an LU solver [106] is possible, it proves too slow even for images of size 256×256. This supports the idea that iterative solvers are necessary for larger images. They use a modified successive over-relaxation (SOR) method [206] with under-relaxation and randomised traversal of mask points. However, storing $\boldsymbol{R}$ becomes impractical for large images due to memory limitations, and recomputing individual inpainting echoes during each iteration is computationally prohibitive. Our method circumvents these issues by efficiently computing matrix-vector products $\boldsymbol{Rx}$ without explicitly forming or storing $\boldsymbol{R}$.

Green's Functions. Homogeneous diffusion inpainting can also be formulated as a linear combination of Green's functions associated with the mask points [111, 113, 125]. These Green's functions serve as a spectral analogue to the inpainting echoes. We denote the matrix of Green's functions for a given mask by $\boldsymbol{G}$, leading to normal equations of the form $\boldsymbol{G}^\top\boldsymbol{G}$ [111]. A key advantage of this formulation is that products $\boldsymbol{Gx}$ can be computed efficiently and without explicitly forming $\boldsymbol{G}$, using fast orthogonal DCT-II and DCT-III transforms [198]. Hoffman [111] constructs $\boldsymbol{G}^\top\boldsymbol{G}$ explicitly and applies a Cholesky decomposition, showing that for a small number of mask points, this approach can outperform conjugate gradient methods used in the classical formulation. However, this method does not scale well to larger images, making it unsuitable for our purposes.

In the 1D setting, Plonka et al. [194] proposed a Green's function-based method that jointly optimises both the mask pixel locations and their tonal values. While effective in one dimension, this approach is not applicable to our 2D image setting and shares the same scalability limitations as Hoffman's method due to its reliance on dense matrices.

Non-binary Mask Optimisation. Hoeltgen and Weickert [108] demonstrated that thresholded non-binary mask spatial optimisation [35, 57, 109, 178] can be interpreted as a joint tonal and binary mask spatial optimisation. Non-binary masks can be optimised with sophisticated non-smooth optimisation strategies such as primal-dual algorithms, which then indirectly perform a tonal optimisation. However, extending these methods to colour images is non-trivial. Moreover, despite their strong performance, these approaches are already computationally demanding even for relatively small images.

Neural-based Approaches. In addition to model-based approaches, neural networks have recently been explored for efficient tonal optimisation. Peter et al. [192] introduced the first neural network specifically designed for tonal optimisation in harmonic inpainting. Their method is both memory-efficient and fast. Although the network is trained to minimise the mean squared error (MSE), it does not explicitly solve the underlying least squares problem, and therefore may not achieve the optimal MSE. Moreover, the network was trained only on low-resolution images and exhibits poor performance on high-resolution inputs such as 4K images. A naïve extension to 4K images would be impractical due to memory and computational limitations of the training process.

Localisation-based Acceleration. Since the influence zone of a single mask pixel is often limited for many inpainting operators, localisation strategies can be employed to accelerate tonal optimisation. These strategies are particularly effective for perfectly localised inpainting operators, such as Shepard interpolation with truncated Gaussians [1, 187] or smoothed particle hydrodynamics [68]. For less local operators, artificial localisation can be introduced through segmentation techniques [112, 119]. Similarly, in homogeneous diffusion inpainting, the support of inpainting echoes can be restricted [111], resulting in an approximate matrix $\tilde{\boldsymbol{R}}$ that is no longer fully dense. However, even with such approximations, the number of restricted echoes that must be computed and stored remains substantial for high-resolution images.

Quantisation. In image compression, quantisation limits the number of available grey or colour values. Incorporating quantisation directly into the tonal optimisation process typically yields better visual quality than applying it as a post-processing step. Several methods have been proposed to address this discrete optimisation problem, including projection-based techniques [191] and stochastic strategies [162, 209]. However, since our work focuses exclusively on the continuous optimisation problem, we do not incorporate these approaches. Nevertheless, in a full compression pipeline, such methods could be readily integrated to enhance performance.

Error Balancing. An early precursor to tonal optimisation was introduced by Galić et al. [89] in 2008. Their method adjusts the tonal values of each mask pixel based on the inpainting error observed in neighbouring pixels. This not only helps to mitigate the impact of singularities at mask locations but also enhances overall inpainting quality. We have extended this concept both theoretically and practically, using it as an efficient strategy to obtain a high-quality initialisation for our tonal optimisation algorithm.

4.1.3 The Linear Least Squares (LLS) Problem

We assume that we are working with the linear inpainting problem

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})\boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}, \tag{4.6}$$

or its generalisation to arbitrary linear features that can be written in terms of linear equality constraints

$$(\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})\boldsymbol{u} = \boldsymbol{P}\boldsymbol{f}, \quad \boldsymbol{P} = \boldsymbol{A}^{+}\boldsymbol{A} \iff \begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^{\top} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{A}\boldsymbol{f} \end{bmatrix}. \tag{4.7}$$

In the spatial optimisation problem, the goal is to replace the values $\boldsymbol{P}\boldsymbol{f}$ with a variable $\boldsymbol{P}\boldsymbol{g}$ giving us the reconstruction as a function of $\boldsymbol{P}\boldsymbol{g}$ (we assume that $\boldsymbol{C}$ or $\boldsymbol{P}$ is fixed)

$$\boldsymbol{u}(\boldsymbol{P}\boldsymbol{g}) = \boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g}, \quad \boldsymbol{B} = \boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}. \tag{4.8}$$

We can now write the tonal optimisation problem, with respect to the Euclidean norm, as a linear least squares problem

$$\min_{\boldsymbol{g}\in\mathbb{R}^N} \|\boldsymbol{u}(\boldsymbol{P}\boldsymbol{g}) - \boldsymbol{f}\|_2^2 = \min_{\boldsymbol{g}\in\mathbb{R}^N} \|\boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g} - \boldsymbol{f}\|_2^2. \tag{4.9}$$

Taking the gradient with respect to $\boldsymbol{g}$ and setting it to zero yields the normal equations

$$\boldsymbol{P}^{\top}\boldsymbol{B}^{-\top}\boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g} = \boldsymbol{P}^{\top}\boldsymbol{B}^{-\top}\boldsymbol{f}. \tag{4.10}$$

The system matrix is generally dense because of the inverses involved, so a direct factorisation of the above is feasible only for relative small images. The idea in [58] is that while the inverses are dense, the matrix $\boldsymbol{B} = (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})$ is sparse, so we could apply an iterative solver whenever we need to compute products with the inverse, which allows us to avoid storing a very large matrix. Such products arise in each step, if we apply an iterative solver to the normal equations (4.10). Thus an efficient approach is to have nested solvers – the outer solving the tonal optimisation problem, while the inner solves the inpainting-like problems for the products with the matrices $\boldsymbol{B}^{-1} = (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})^{-1}$ and $\boldsymbol{B}^{-\top} = (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})^{-\top}$.

FEM interpolation. In FEM, the system in (4.6) gives us only the values at the vertices. We additionally have to interpolate over all image pixels in order to get the full reconstruction. This should be accounted for in the tonal optimisation formulation. Let the interpolation matrix be $\boldsymbol{Y} \in \mathbb{R}^{N \times n}$, where Y_{ij} is the j-th FEM trial function evaluated at the i-th image pixel, then the FEM inpainting has the form

$$\boldsymbol{u}(\boldsymbol{P}\boldsymbol{g}) = \boldsymbol{Y}\boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g}. \tag{4.11}$$

Consequently, the normal equations become

$$\boldsymbol{P}^\top\boldsymbol{B}^{-\top}\boldsymbol{Y}^\top\boldsymbol{Y}\boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g} = \boldsymbol{P}^\top\boldsymbol{B}^{-\top}\boldsymbol{Y}^\top\boldsymbol{f}. \tag{4.12}$$

Since for $\mathbb{P}_1$ elements the basis functions have overlaps only for the one-ring neighbours of a vertex, $\boldsymbol{Y}$ is sparse and we can efficiently evaluate products with $\boldsymbol{Y}^\top\boldsymbol{Y}$.

4.1.4 Krylov Methods for the LLS Problem

One of the most efficient classes of linear system solvers are Krylov subspace methods. A natural solver for linear least squares problems is the conjugate gradient for the normal equations (CGNR), see Chapter 8 in [206] or Chapter 7.4 in [32], which monotonically decreases the 2-norm of the residual in each step

$$\|\boldsymbol{r}(\boldsymbol{P}\boldsymbol{g})\|_2 = \|\boldsymbol{u}(\boldsymbol{P}\boldsymbol{g}) - \boldsymbol{f}\|_2. \tag{4.13}$$

It is equivalent to the conjugate gradient solver applied to the normal equations, but rewritten so that it is more numerically stable, since the normal equations square the condition number of the matrix in the minimisation problem and can lead to numerical issues. A natural stopping criterion for the tonal optimisation problem is then based on the percentage decrease per iteration of the squared Euclidean norm of the residual

$$\|\boldsymbol{r}^{k+1}\|_2^2 > (1 - \mathrm{tol})\|\boldsymbol{r}^k\|_2^2. \tag{4.14}$$

For example we may require that the squared 2-norm (equivalent to the MSE up to a constant) decreases by at least tol = 1% or tol = 0.1% per iteration. Generally, the more ill-conditioned our inpainting matrix, the smaller we should set tol, as for more ill-conditioned systems there are many steps that yield only a small decrease. We take a minimum of 1% decrease for harmonic inpainting and 0.1% for biharmonic inpainting.

4.1.5 Krylov Methods for the Matrix-Vector Products

As we emphasised in Section 4.1.3, ideally we should not factorise or invert the inpainting matrix, but instead use iterative solvers to evaluate the matrix-vector products. This allows us to leverage the sparsity of the inpainting matrix.

If $\boldsymbol{L}$ is symmetric, we can also rewrite the inpainting system to be symmetric:

$$\begin{aligned}(\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})\boldsymbol{u} &= \boldsymbol{P}\boldsymbol{g}\\ (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{I} - \boldsymbol{P}))\boldsymbol{u} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}\boldsymbol{P}\boldsymbol{u} &= \boldsymbol{P}\boldsymbol{g}\\ (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{I} - \boldsymbol{P}))\boldsymbol{u} &= (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L})\boldsymbol{P}\boldsymbol{g},\end{aligned} \tag{4.15}$$

where we have used that $\boldsymbol{Pu} = \boldsymbol{Pg}$ in order to go from the second to the third equation. For brevity of notation we define the following matrices:

$$\tilde{\boldsymbol{B}} = (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{I} - \boldsymbol{P})), \quad \tilde{\boldsymbol{P}} = \boldsymbol{P} - (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{LP}. \tag{4.16}$$

If $\boldsymbol{L}$ is symmetric then $\tilde{\boldsymbol{B}}$ is symmetric, and we can write the normal equations as follows

$$\tilde{\boldsymbol{P}}^{\top}\tilde{\boldsymbol{B}}^{-2}\tilde{\boldsymbol{P}}\boldsymbol{g} = \tilde{\boldsymbol{P}}^{\top}\tilde{\boldsymbol{B}}^{-1}\boldsymbol{f}. \tag{4.17}$$

Here all inverses involve the symmetrised inpainting matrix

$$\tilde{\boldsymbol{B}} = (\boldsymbol{P} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{I} - \boldsymbol{P})). \tag{4.18}$$

If the latter is positive semi-definite we can apply the conjugate gradient (CG) [206] solver to approximately compute products with its (pseudo-)inverse

$$\boldsymbol{q} = \tilde{\boldsymbol{B}}^{+}\boldsymbol{p} \iff \tilde{\boldsymbol{B}}\boldsymbol{q} = \boldsymbol{p}. \tag{4.19}$$

If the matrix $\boldsymbol{L}$ is symmetric but indefinite we can use the conjugate residual (CR) [206], SYMMLQ [180], or the MINRES solver [86, 180]. For non-symmetric $\boldsymbol{L}$, solvers such as the bi-conjugate gradient stabilised (Bi-CGSTAB) [233] can be employed.

When we consider linear features inpainting, for efficiency reasons we should not form $\boldsymbol{P}$ and should instead use the saddle-point system for the inpainting

$$\begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^{\top} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Ag} \end{bmatrix}. \tag{4.20}$$

The latter is automatically symmetric if $\boldsymbol{L}$ is symmetric, but it is indefinite, so a CR solver (see 6.8 in [206]) should be used as discussed in Section 2.3.3. This concludes the theory for the tonal optimisation approaches that were applied in [58] and [120].

4.1.6 Initial Guess for Tonal Optimisation

Iterative solvers can significantly benefit from a well-chosen initial iterate. Therefore, we briefly review the strategy proposed in [128] for constructing such an initial guess, and then extend it to the feature inpainting setting.

Error Balancing. The method in [128] builds upon the approach by Galić et al. [89], which does not address the full tonal optimisation problem but instead solves a simplified surrogate problem. The core idea is to adjust the tonal value of each mask pixel by incorporating the average signed inpainting error of its neighbours:

$$\boldsymbol{g}_i = \boldsymbol{u}_i + \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} (\boldsymbol{f}_j - \boldsymbol{u}_j) \quad \text{for } c_i = 1, \tag{4.21}$$

where $\mathcal{N}(i)$ denotes all direct neighbours of the mask pixel i, including diagonal neighbours and the pixel itself.

Interpolation of Local Averages. We interpret the method described above as performing a single step toward making the mask pixels interpolate the averages over their neighbourhoods. This perspective is closely related to the local average constraints introduced in [120].

Building on this interpretation, we generalise the method by introducing a weight matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N}$ to define the averaging around the mask pixels. Instead of interpolating the values of $\boldsymbol{f}$ directly at the mask pixels, we aim to interpolate the local averages $(\boldsymbol{W}\boldsymbol{f})_i$, leading to the following constraint:

$$\boldsymbol{C}\boldsymbol{W}\boldsymbol{B}^{-1}\boldsymbol{C}\boldsymbol{g} =: \boldsymbol{C}\boldsymbol{W}\boldsymbol{u} = \boldsymbol{C}\boldsymbol{W}\boldsymbol{f}. \tag{4.22}$$

We solve equation (4.22) by using a modified Richardson iteration:

$$\boldsymbol{C}\boldsymbol{g}^{\ell+1} = \boldsymbol{C}\boldsymbol{g}^{\ell} + \tau(\boldsymbol{C}\boldsymbol{f} - \boldsymbol{C}\boldsymbol{W}\boldsymbol{B}^{-1}\boldsymbol{C}\boldsymbol{g}^{\ell}). \tag{4.23}$$

In our framework, the original method by Galić et al. corresponds to a single Richardson step with step size $\tau = 1$ and a simple choice of $\boldsymbol{W}$. By performing multiple iterations, we can further reduce the MSE.

We choose τ such that the scheme remains stable in the 2-norm, i.e., $\|\boldsymbol{C} - \tau\boldsymbol{C}\boldsymbol{W}\boldsymbol{A}^{-1}\boldsymbol{C}\|_2 < 1$. The iteration is terminated once the MSE begins to increase. This is justified because the average interpolation problem is only an approximate surrogate for the tonal optimisation problem and not equivalent to it.

Voronoi Diagram Average. To obtain a closer surrogate to the tonal optimisation problem, we propose a more accurate approximation of each mask pixel's influence zone. The cells of the Voronoi diagram induced by the mask provide a reasonable estimate of the regions affected by the inpainting echoes. Therefore, we suggest replacing the direct neighbours in $\boldsymbol{W}$ with the neighbours defined by the Voronoi cell of each mask pixel.

Extension to Feature Inpainting. We go beyond our work in [120] and [128], and extend this initial guess to the setting of the features inpainting framework. The main idea remains the same except for the fact that we have multiple mask matrices $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_k$ for the k families of features $\boldsymbol{F}_1, \ldots, \boldsymbol{F}_k$, such that $\boldsymbol{A}_j = \boldsymbol{C}_j\boldsymbol{F}_j$. We then require that

$$\boldsymbol{A}_j\boldsymbol{W}\boldsymbol{u}(\boldsymbol{g}) = \boldsymbol{A}_j\boldsymbol{W}\boldsymbol{f}, \quad 1 \leq j \leq k. \tag{4.24}$$

This Richardson update is overdetermined for $\boldsymbol{g}$. One option is to use instead

$$\boldsymbol{P}\boldsymbol{W}\boldsymbol{u}(\boldsymbol{g}) = \boldsymbol{P}\boldsymbol{W}\boldsymbol{f} \implies \boldsymbol{P}\boldsymbol{g}^{\ell+1} = \boldsymbol{P}\boldsymbol{g}^{\ell} + \tau(\boldsymbol{P}\boldsymbol{f} - \boldsymbol{P}\boldsymbol{W}\boldsymbol{B}^{-1}\boldsymbol{P}\boldsymbol{g}^{\ell}). \tag{4.25}$$

Table 4.1: Runtime scaling with resolution of our tonal optimisation for optimised 4% masks. The reported times are for a system with a *Ryzen 4800H* CPU.

image size	64 × 64	128 × 128	256 × 256	512 × 512	1024 × 1024
tonal opt.	0.03s	0.14s	0.53s	2.96s	12.77s

Another option is for us to modify the inpainting as follows

$$\begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{A}\boldsymbol{g} \end{bmatrix} \to \begin{bmatrix} \boldsymbol{L} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{h} \end{bmatrix}. \tag{4.26}$$

That is, we have replaced $\boldsymbol{A}\boldsymbol{g}$ by $\boldsymbol{h}$. The latter is more in line with an approximation framework than an interpolating one. Then the interpolation condition becomes

$$\boldsymbol{A}_j \boldsymbol{W} \boldsymbol{u}(\boldsymbol{h}) = \boldsymbol{A}_j \boldsymbol{W} \boldsymbol{f}, \quad 1 \le j \le k, \tag{4.27}$$

which has the following Richardson update rule:

$$\boldsymbol{C}_j \boldsymbol{h}^{\ell+1} = \boldsymbol{C}_j \boldsymbol{h}^{\ell} + \tau\big(\boldsymbol{A}_j \boldsymbol{f} - \boldsymbol{A}_j \boldsymbol{W} \boldsymbol{u}(\boldsymbol{h}^{\ell})\big), \quad 1 \le j \le k. \tag{4.28}$$

Practical Aspects. The construction of the initial guess typically incurs only half the cost per iteration compared to the Krylov solver. Moreover, it requires very few iterations, in which we see the greatest MSE improvements. As such it is more efficient than running the Krylov solvers from a standard initial iterate that interpolates the data. In fact, if time is a concern one can drop the Krylov optimisation altogether and use just the initial iterate as a cheaper way to get a tonally optimised image of a slightly worse quality.

We note that the method is general and is not necessarily restricted to the harmonic inpainting setting discussed in [128]. We have applied it with much success in the biharmonic setting.

4.1.7 Experiments: FEM Harmonic Inpainting

In Table 4.1 we demonstrate that our tonal optimisation method from [58] scales linearly with the image size. This showcases the efficiency of our nested Krylov solver approach. Note that in general factorisation methods would scale as $O(N^3)$, and one would very quickly run out of memory as the image size grows.

On the other hand, in Table 4.2 we showcase the qualitative improvements that tonal optimisation provides on optimised masks. We note that the MSE decrease can be almost 30%. The left column corresponds to the number of densification iterations (the more iterations the more work the spatial optimisation does).

Table 4.2: MSE comparisons at 4% density without and with tonal optimisation.

Method	*trui*		*walter*		*peppers*	
	no TO	with TO	no TO	with TO	no TO	with TO
Ours ($n = 10$)	44.62	30.07	19.09	12.62	43.20	29.83
Ours ($n = 30$)	40.58	28.21	16.35	**11.09**	38.37	**28.11**
Ours ($n = 100$)	**37.60**	**26.62**	**15.92**	11.21	**36.68**	28.85

4.1.8 Experiments: Initial Guess and Scaling

In this subsection we reproduce plots from our experiments in [128]. In Fig. 4.1 we illustrate the effect of the initial guess, denoted as VI (Voronoi initialisation) – note that the red curve starts much higher than the rest thanks to the initial guess, and this is achieved in a fraction of the total time. In the figure, RAS refers to restricted additive Schwarz applied to the tonal optimisation problem, which we do not discuss here. It suffices to say that a Voronoi initialised CGNR solver is generally not far behind RAS+VI. In Fig. 4.2 we see that the runtime-to-resolution scaling is linear – similar to our FEM experiments. The curves are the result of averaged metrics from experiments on a set of twelve natural 4K images. The experiments are from our journal publication [128] and were conducted on a system with an *AMD Ryzen 5900X@3.7GHz* CPU and an *Nvidia GeForce GTX 1080 Ti* GPU. Notably, the tonal optimisation and inpainting were implemented on the GPU.

4.1.9 Experiments: Linear Feature Inpainting

Figure 4.2 demonstrates the effect of our tonal optimisation for feature inpainting. When applied to spatially optimised representations using five feature types (colours, x and y derivatives, and two blurring kernels), it reduces the mean squared error (MSE) by approximately one third. It is also important to note that optimising the function values does not increase the data volume – high-quality values require no more storage than poor ones.

4.1.10 Conclusion

We developed an efficient tonal optimisation algorithm that also extends to linear feature inpainting and FEM. The algorithm relies on nested Krylov solvers, which allowed us to exploit the sparsity of the inpainting matrices, and circumvented the need of computing and storing dense matrix factorisations. We empirically verified that our approach scales linearly in terms of the image resolution and that it provides about a 30% decrease of the MSE when applied to harmonic inpainting (including feature inpainting). An interesting avenue for future work is the extension of the above method to nonlinear inpaintings such as EED, and potentially also to nonlinear feature inpainting. One could apply a Gauss-Newton, Levenberg-Marquardt, or a full Newton approach there. The main

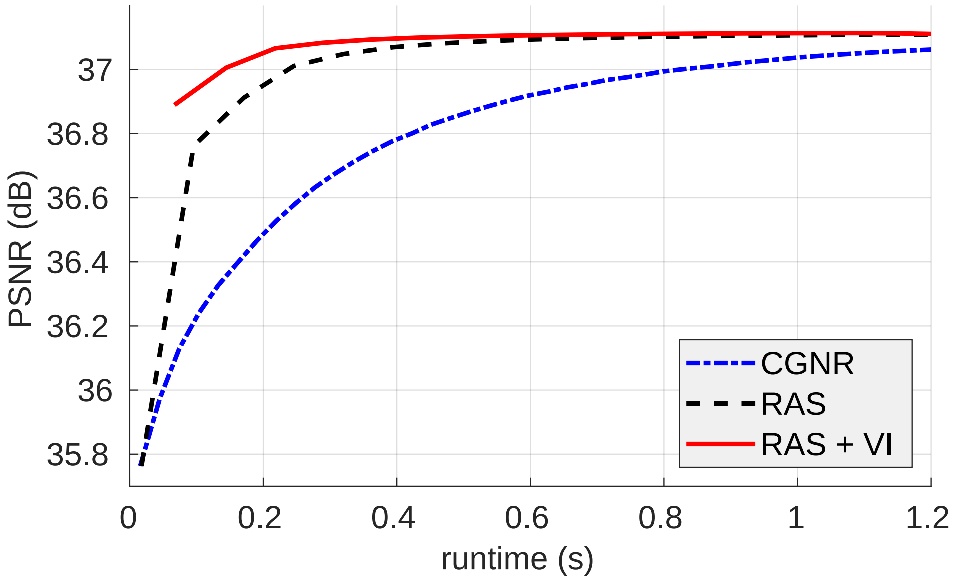


FIGURE 4.1: Effect of the initial guess for 5% mask densities. Plot courtesy of Niklas Kämper.

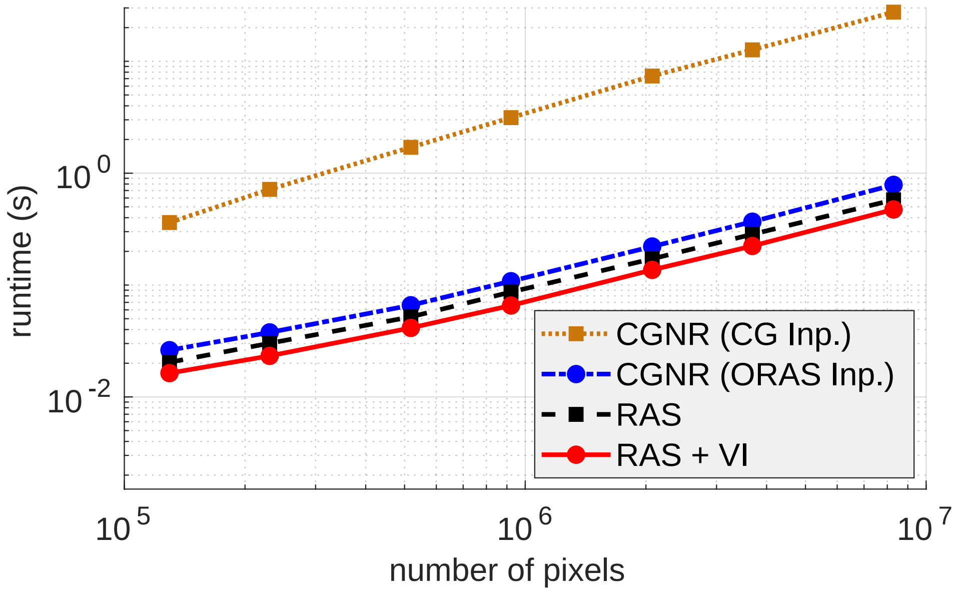
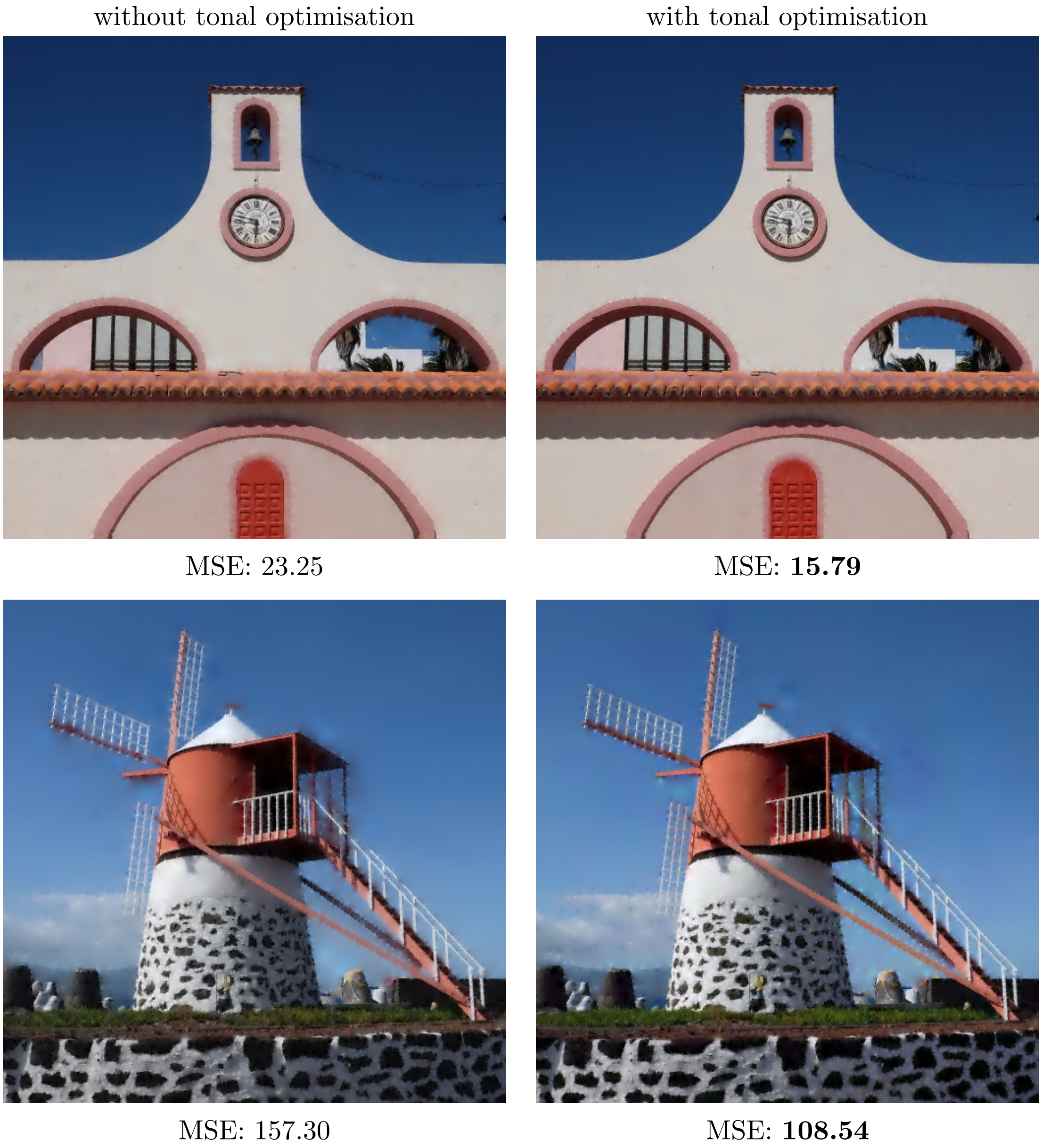


FIGURE 4.2: Runtime depending on the number of pixels (double logarithmic plot). Plot courtesy of Niklas Kämper.

challenge is obviously the implementation of efficient solvers. It should also be mentioned that the initialisation that we discussed can directly be applied to the nonlinear setting without major modifications.



without tonal optimisation

with tonal optimisation

MSE: 23.25

MSE: **15.79**

MSE: 157.30

MSE: **108.54**

Figure 4.2: Test images *elpaso* and *windmill* and their sparse representations using all five proposed feature types from Section 2.5.1 (total mask density: 5%) without and with tonal optimisation. The tonal optimisation improves the reconstruction quality by about one third. Note, however, the localised emphasis of the artifacts – this is also a general drawback observed with tonal optimisation for classical harmonic inpainting – it emphasises the logarithmic singularities, and is inherent to the inpainting operator.

4.2 Spatial Optimisation

In the current section we describe our proposed algorithms from [58, 120, 128], and extensions thereof, for approximately solving the spatial optimisation problem for interpolating inpainting

$$\min_{\|\boldsymbol{c}\|_0 = m} E(\boldsymbol{c}), \quad E(\boldsymbol{c}) = \|\boldsymbol{u}(\boldsymbol{c}, \boldsymbol{C}\boldsymbol{f}) - \boldsymbol{f}\|, \tag{4.29}$$

or in the features inpainting setting [120] the more general problem

$$\min_{\|\boldsymbol{c}\|_0 = m} E(\boldsymbol{c}), \quad E(\boldsymbol{c}) = \|\boldsymbol{u}(\boldsymbol{c}, \boldsymbol{C}\boldsymbol{F}\boldsymbol{f}) - \boldsymbol{f}\|, \quad \boldsymbol{A} = \boldsymbol{C}\boldsymbol{F}. \tag{4.30}$$

Unlike in the tonal optimisation setting, we do not require that $\boldsymbol{u}$ is linear – our algorithms are oblivious to this. Moreover, we can also allow nonlinear features, by considering the problem

$$\min_{\|\boldsymbol{c}\|_0 = m} E(\boldsymbol{c}), \quad E(\boldsymbol{c}) = \|\boldsymbol{u}(\boldsymbol{c}, \boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{f})) - \boldsymbol{f}\|, \tag{4.31}$$

where the function $\boldsymbol{\phi} : \mathbb{R}^N \to \mathbb{R}^{k \cdot N}$ describes the considered families of (nonlinear) features

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi}_1^\top & \cdots & \boldsymbol{\phi}_k^\top \end{bmatrix}^\top, \quad \boldsymbol{c} = \begin{bmatrix} \boldsymbol{c}_1^\top & \cdots & \boldsymbol{c}_k^\top \end{bmatrix}^\top. \tag{4.32}$$

Since our algorithm for the nonlinear setting subsumes the linear case, we directly state its most general version.

4.2.1 Outline

We review related work in Section 4.2.2. This is followed, in Section 4.2.3, by an exposition of the class of greedy strategies that we base our algorithms on. The latter rely on an oracle, which we discuss in Section 4.2.4. The algorithms also use a partitioning of the domain, in order to avoid clustering and improve the oracle. We discuss the partitioning in Section 4.2.5. We present our Voronoi densification algorithm in Section 4.2.6. Then, in Section 4.2.7, we study the connection between our approach and matching pursuit algorithms. We empirically validate our spatial optimisation algorithm for FEM harmonic inpainting in Section 4.2.8. In Section 4.2.9, we extend the experiments to the setting of nonlinear feature inpainting. We conclude our discussion of spatial optimisation in Section 4.2.10.

4.2.2 Related Work

Finding a subset of pixels to be stored that result in a high quality reconstruction is a challenging combinatorial optimisation problem. Here we only focus on approaches for sparse image inpainting in 2D. There are also a number of related works, for example the free knot problem for spline interpolation [214].

Analytic Approach. Belhachmi et al. [28] introduced a framework for mask optimisation based on a continuous shape-analytic interpretation of homogeneous diffusion inpainting. This approach was later extended by Belhachmi and Jacumin [25, 27] to handle noisy images. A key practical insight from their work is that the mask density should increase with the absolute value of the Laplacian of the target image. In the discrete setting, they propose generating such masks by applying dithering – such as Floyd–Steinberg dithering [85] – to the absolute value of a smoothed Laplacian image. This method is computationally efficient, as it avoids solving any inpainting problems. However, the dithering approximation often results in limited reconstruction quality [156]. Our approach achieves significantly higher quality reconstructions, albeit at a higher cost. We can, however, initialise our densification with the result from the analytic approach in the first iterations, so that our method gracefully transitions into the analytic approach if zero densification iterations are chosen.

Gradient-based Methods. High-quality non-binary inpainting masks can be obtained using non-smooth optimisation techniques such as primal-dual solvers and optimal control methods [35, 57, 109, 178]. However, for compression applications, these masks must ultimately be binarised, which typically leads to a reduction in reconstruction quality [108]. Despite this drawback, such methods still produce state-of-the-art results. Their main limitation lies in their computational cost, as they are prohibitively slow even for small images. Since our goal is to develop fast algorithms that scale efficiently to large resolutions, we instead focus on greedy strategies that offer a favourable trade-off between quality and runtime.

Sparsification. Mainberger et al. [156] proposed the probabilistic sparsification (PS) algorithm, which begins with a fully populated mask and iteratively removes pixels until the desired mask density is achieved. In each iteration, a subset of candidate pixels is temporarily removed from the mask, and an error map is computed based on the resulting inpainting. A fixed number of candidates – those with the highest pointwise reconstruction errors – are then reinserted into the mask. This method is flexible and can be adapted to various inpainting operators, including PDE-based approaches [156] and linear spline interpolation over triangulations [70]. However, it is computationally expensive. Moreover, because PS starts from a full mask, achieving low densities requires many iterations, making it less efficient than densification strategies that begin with an empty mask. Finally, PS is susceptible to suboptimal local minima due to its reliance on pointwise error as an oracle and the limited predictive power of early-stage errors, which are influenced only by local neighbourhoods. These limitations motivate the exploration of densification strategies guided by oracles that adapt dynamically to the current mask density.

Densification. Densification approaches begin with an empty mask and iteratively insert new pixels. This strategy typically requires fewer iterations than sparsification-based methods, especially in compression scenarios where target mask densities are usually below 10%. In compression contexts, densification has been effectively applied to constrained data structures such as subdivision trees [72, 89, 191, 209]. In our work, however, we focus on the more general case of unconstrained masks. Unconstrained densification strategies have been successfully employed in various other settings, including diffusion-based [26, 68, 229] and exemplar-based [131] inpainting, as well as linear interpolation on Delaunay triangulations [3, 80]. Notably, the work of Daropoulos et al. [68] as well as our works [58, 120, 128] go beyond pixel-wise error metrics by aggregating errors over mask-adaptive neighbourhoods defined by Voronoi diagrams [68, 120, 128] or Delaunay triangulations [58, 128]. This leads to improved reconstruction quality. Currently, Voronoi- and Delaunay-based densification strategies offer the best trade-off between reconstruction quality and runtime for mask construction. As such, we adopted these approaches as the foundation for the mask generation algorithms developed in our work.

Non-local Pixel Exchange. Although densification methods often outperform sparsification ones, they remain inherently greedy and are therefore susceptible to getting trapped in local minima. To address this limitation, Mainberger et al. [156] introduced a global relocation strategy known as non-local pixel exchange (NLPE). This method stochastically selects a mask pixel and a subset of non-mask pixels with high pointwise reconstruction errors. It then evaluates the reconstruction quality resulting from relocating the selected mask pixel to each candidate position, retaining the best swap (which may be none). While NLPE can effectively escape local minima and yield substantial improvements, it requires a large number of iterations to converge, making it computationally expensive. A similar relocation strategy has also been applied to optimise masks for linear spline interpolation over triangulations [162].

Neural-based Approaches. In recent years, several neural frameworks have been proposed for mask generation, offering an alternative to model-based spatial optimisation. Dai et al. [65] introduced a deep learning approach for adaptive sampling, employing separate networks for inpainting and mask optimisation. To improve reconstruction quality, Peter [188] jointly trains inpainting and spatial optimisation with Wasserstein GANs. Alt et al. [9] focused specifically on homogeneous diffusion inpainting, training a mask generator alongside a learned surrogate solver that approximates the inpainting process, enabling efficient backpropagation. The efficiency has been improved further by Peter et al. [192] with a modified network architecture.

However, these neural networks were trained only on low-resolution images, resulting in poor performance on higher-resolution inputs. Direct training on high-resolution data is infeasible on standard hardware due to memory and computational constraints. To address this, Schrader et al. [213] proposed a coarse-to-fine strategy that divides the image into smaller patches. Each patch is assigned a mask by a neural network, with the local mask density guided by the average Laplacian magnitude, following the analytic approach of Belhachmi et al. [28]. This enables efficient scaling to high resolutions, allowing mask generation for 4K colour images in under 0.5 seconds on high-end GPUs.

Video Coding. Most approaches to inpainting-based video compression, such as [14, 190], optimise the inpainting masks independently for each frame. In contrast, Breuß et al. [39] propose a more temporally coherent strategy by optimising masks only for key frames using the optimal control framework of Hoeltgen et al. [109]. For intermediate frames, they employ optical flow to interpolate the masks, thereby reducing computational cost while maintaining temporal consistency.

4.2.3 Densification and Sparsification

An efficient approach to tackling the spatial optimisation problem is based on iteratively adding or removing new mask points over multiple iterations. In densification approaches we start with an empty mask, and at each iteration we aim to add mask points with locations chosen such that the error is decreased the most. In sparsification approaches we start from a full mask, and aim to remove mask points in each iteration, such that the error increases the least. Generally, densification approaches are a more natural choice if we aim at a mask density under 50%, while sparsification approaches are more efficient for mask densities over 50%. There are, however, other considerations, such as the fact that adding points can usually localise the effect of previous points, while removing points makes the effect of old points more global. Taking this into account and also the fact that we target low densities, we choose to use a densification method.

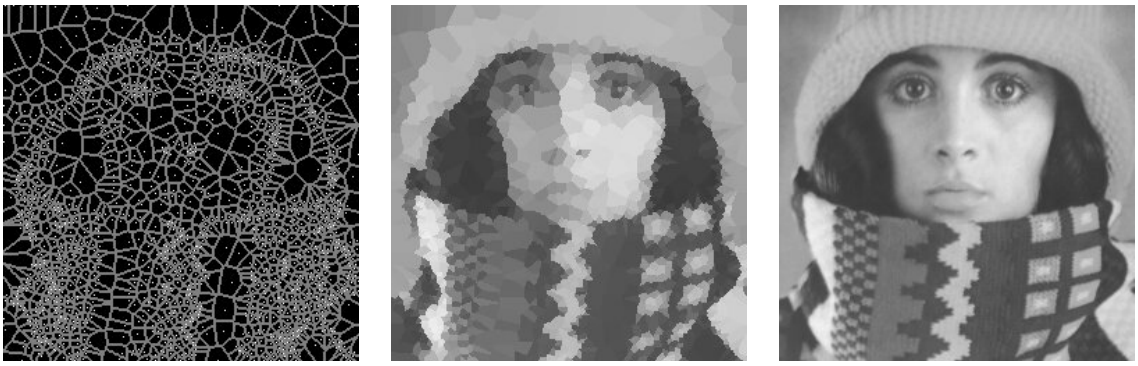
Computational Cost of Full Densification (Sparsification). In the extreme case we can add (remove) a single point per iteration and evaluate the error for all possible candidates. This means that in step k we need to perform as many inpaintings as there are zeros (ones) in $\boldsymbol{c}^{k-1}$ in order to evaluate the error. When summed over all steps, this yields a total of $m!$ (sparsification: $(N-m)!$) number of inpaintings. To reduce this, we add (remove) multiple mask points per iteration, and evaluate the inpainting only at some locations instead of all possible locations. Furthermore, instead of computing an inpainting for all of the candidates and recomputing the MSE, we use an oracle that tries to predict which are the best candidates. We discuss the oracle in Section 4.2.4 and a way to avoid clustering when adding multiple mask points in Section 4.2.5.

4.2.4 Error Map

After each densification iteration we evaluate the signed error $\boldsymbol{e}^k = \boldsymbol{u}^k - \boldsymbol{f}$ in order to try and predict at what locations new mask points should be introduced. Notably, if we normalise the rows of the matrix $\boldsymbol{F}^k = \boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k)$ (where $\boldsymbol{\phi}$ are the features) w.r.t. the 2-norm, the product $\boldsymbol{F}^k \cdot \boldsymbol{e}^k$ gives us the dot products between the (linearisation of the) features and the error. In the nonlinear setting this is motivated by the linearisation

$$\boldsymbol{\phi}(\boldsymbol{f}) = \boldsymbol{\phi}(\boldsymbol{u}^k - \boldsymbol{e}^k) \approx \boldsymbol{\phi}(\boldsymbol{u}^k) - \boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k) \cdot \boldsymbol{e}^k \implies \boldsymbol{\phi}(\boldsymbol{u}^k) - \boldsymbol{\phi}(\boldsymbol{f}) \approx \boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k) \cdot (\boldsymbol{u}^k - \boldsymbol{f}). \quad (4.33)$$

That is, we can use $\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k)(\boldsymbol{u}^k - \boldsymbol{f})$ as an oracle in order to figure out where to introduce a new point such that $\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^{k+1})(\boldsymbol{u}^{k+1} - \boldsymbol{f}) \approx \boldsymbol{\phi}(\boldsymbol{u}^{k+1}) - \boldsymbol{\phi}(\boldsymbol{f})$ would be reduced. For



Voronoi tessellation piecewise-constant approx. original image *trui*

FIGURE 4.3: A visualisation of the Voronoi tessellation induced by a mask, a piecewise-constant approximation based on the Voronoi cells, and the original image.

linear features ≈ is of course an equality. We can expect that at locations where the magnitude of these inner products is large, the specific feature can compensate well for the error. This is similar in nature to the sensing step in matching pursuit [158, 200] or thresholding [210], however, in our case we do not have the atoms giving the true reconstruction. As such the magnitudes of the elements of $\boldsymbol{F}^k \cdot \boldsymbol{e}^k$ typically underestimate the error decrease – notably they ignore the additional decrease in a local neighbourhood due to the inpainting operator. Moreover, if we try to introduce multiple points per iteration, there is a danger that they cluster due to trying to reduce the same error. We mitigate both of these issues in the next subsection.

4.2.5 Partition-based Densification

In order to disallow clustering when introducing multiple points, we partition the domain and allow only a single point to be introduced per region. The partition should be non-overlapping by definition and it should ideally be adapted to the current mask structure. A simple partition with a multitude of desirable properties is given by the Voronoi tessellation [18] – a visualisation of such a partition adapted to a mask is shown in Figure 4.3. Then to handle the clustering problem we allow introducing only a single point per cell. In order to mitigate the problem of $\boldsymbol{F} \cdot \boldsymbol{e}$ underestimating the error decrease, we can integrate the squared error in each cell and choose to introduce new mask points only in the cells with highest error. This works under the assumption that locally the inpainting operator is able to decrease the error inside the cell. Within the cell we add the mask point at index i where $|(\boldsymbol{F} \cdot \boldsymbol{e})_i|$ is the largest.

4.2.6 Voronoi Densification for Equality Constrained Features

Putting all of these ideas together leads us to a more general and improved version of the algorithm from our previous works [58, 120, 128]. In it we employ a densification strategy with an inpainting at each iteration, an error map computed from the latter, and a Voronoi decomposition. The main difference is the support for nonlinear features. We also now normalise the rows of $\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k)$ w.r.t. the 2-norm instead of the 1-norm used in [120]. That is, $\boldsymbol{F}^k = \boldsymbol{D}^k \boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k)$ where $\boldsymbol{D}^k$ is a diagonal matrix with the reciprocals of the 2-norms of the rows of $\boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k)$. Another change from [120] is that we integrate the squared error per cell, instead of the squared features' errors. The listed modifications are more in line with well-founded ideas from matching pursuit algorithms and yield considerable improvements. The approach is described in Algorithm 2. A visual representation of the inner workings of the algorithm is shown in Fig. 4.4.

Algorithm 2. Voronoi Densification for Feature Inpainting

```
Input : Original image $\boldsymbol{g}$, number of iterations $n$, number of desired mask points $m$
Output : Inpainting mask $\boldsymbol{c}$, reconstruction $\boldsymbol{u}$
Initialise: Initial mask $\boldsymbol{c}$ with $\lceil \frac{m}{n} \rceil$ mask pixels
for $i = 1$ to $n - 1$ do
  Construct the Voronoi tessellation $\{\mathcal{T}_j\}$ of the current mask pixels.
  Compute the inpainting $\boldsymbol{u}^k = \boldsymbol{u}(\boldsymbol{c}^k, \boldsymbol{C}^k \boldsymbol{\phi}(\boldsymbol{f}))$ and the error map $\boldsymbol{e}^k = \boldsymbol{u}^k - \boldsymbol{f}$.
  Compute the cells 2-norm errors $\forall j$, $e^k_{\mathcal{T}_j} = \sum_{i \in \mathcal{T}_j} (e^k_i)^2$.
  Find the $\lceil \frac{m}{n} \rceil$ Voronoi cells $\{\mathcal{T}_{j_i}\}_{i=1}^{\lceil \frac{m}{n} \rceil}$ with the highest errors $\{e^k_{\mathcal{T}_{j_i}}\}_{i=1}^{\lceil \frac{m}{n} \rceil}$.
  For each cell in $\{\mathcal{T}_{j_i}\}_{i=1}^{\lceil \frac{m}{n} \rceil}$ find the entry in $\boldsymbol{D}^k \boldsymbol{J}_{\boldsymbol{\phi}}(\boldsymbol{u}^k) \cdot \boldsymbol{e}^k|_{\mathcal{T}_{j_i}}$ with highest magnitude
  and add it to the corresponding location in $\boldsymbol{c}^{k+1}$.
```

4.2.7 Relation to Matching Pursuit

In the literature, matching pursuit algorithms [158, 200] have been used to solve a simpler and somewhat different version of the data optimisation problem, namely

$$\min_{\boldsymbol{a}; \|\boldsymbol{c}\|_0 \leq m} \|\boldsymbol{D}\boldsymbol{C}\boldsymbol{a} - \boldsymbol{f}\|_2^2, \tag{4.34}$$

where $\boldsymbol{D}$ is a dictionary, $\boldsymbol{a}$ are coefficients, and $\boldsymbol{c}$ is the mask. The main difference compared to our problem is that the dictionary is given, and that the reconstruction $\boldsymbol{u} = \boldsymbol{D}\boldsymbol{C}\boldsymbol{a}$ is linear in the coefficients. The above can in fact be reconciled with linear inpainting with colour value interpolation constraints by considering the spectral counterpart of the inpainting problem. That is, if we have an inpainting operator $\boldsymbol{L}$, we may consider its pseudoinverse

$$\boldsymbol{L}^+ = \begin{bmatrix} \boldsymbol{W}_{\text{img}} & \boldsymbol{W}_{\text{ker}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_{11}^{-1} & 0 \\ \boldsymbol{0} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{\text{img}}^\top \\ \boldsymbol{U}_{\text{ker}}^\top \end{bmatrix}, \quad \boldsymbol{L} = \begin{bmatrix} \boldsymbol{U}_{\text{img}} & \boldsymbol{U}_{\text{ker}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_{11} & 0 \\ \boldsymbol{0} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_{\text{img}}^\top \\ \boldsymbol{W}_{\text{ker}}^\top \end{bmatrix}, \tag{4.35}$$

and then any inpainting can be written as

$$\boldsymbol{u} = \boldsymbol{L}^+ \boldsymbol{C}\boldsymbol{a} + \boldsymbol{W}_{\text{ker}} \boldsymbol{\mu}, \quad \boldsymbol{U}_{\text{ker}}^\top \boldsymbol{C}\boldsymbol{a} = \boldsymbol{0}. \tag{4.36}$$

Here $\boldsymbol{L}^+$ plays the role of the dictionary. We also need a part from the kernel if the operator is singular, but the optimal values for the latter can be computed directly as $\boldsymbol{\mu} = \boldsymbol{W}_{\text{ker}}^\top \boldsymbol{f}$, and then we can write the problem in a similar way as the classical

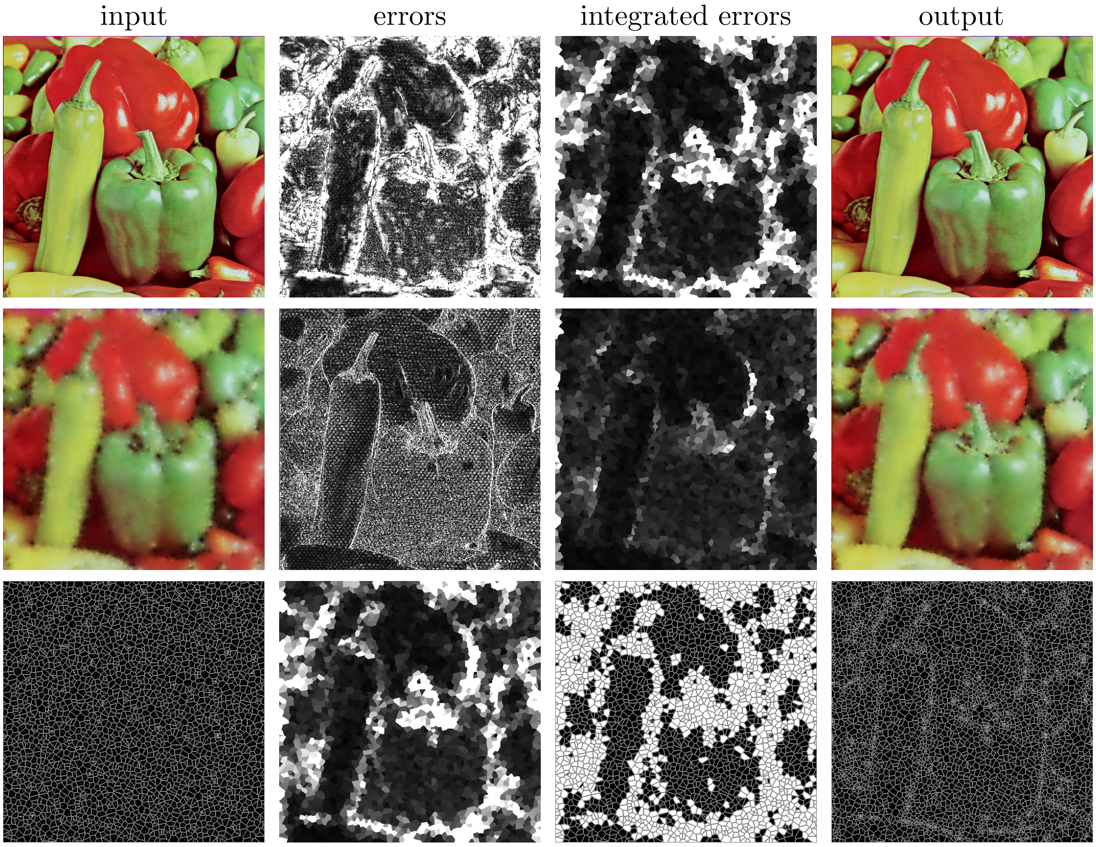


Figure 4.4: Steps in a single iteration of our densification algorithm. The **first column** is the output of the previous step. From top to bottom: reference image, inpainting, mask with Voronoi diagram. The **second column** consists of the error maps for the Dirichlet and ∂_x features, and the maximum cell-wise integrated errors. The **third column** contains cell-wise integrated errors for the Dirichlet and ∂_x features, and the selected $\lceil m/n \rceil$ cells with largest errors that are to be refined (in white). The **last column** shows the output of this iteration: reference image, new inpainting, and updated mask with Voronoi diagram.

formulation:

$$\min_{\substack{\|\boldsymbol{c}\|_0 \leq m \\ \boldsymbol{U}_{\text{ker}}^\top \boldsymbol{C}\boldsymbol{a} = \boldsymbol{0}}} \|\boldsymbol{L}^+ \boldsymbol{C}\boldsymbol{a} - \boldsymbol{W}_{\text{img}} \boldsymbol{W}_{\text{img}}^\top \boldsymbol{f}\|_2^2. \tag{4.37}$$

Applying matching pursuit can work well provided the Moore-Penrose pseudoinverse $\boldsymbol{L}^+$ is not too ill-conditioned. If the matrix is ill-conditioned (which happens to be so even for harmonic inpainting), one typically needs to consider orthogonal matching pursuit which also performs a projection at each step. This has been done in 1D for harmonic and biharmonic inpainting in the work by Plonka et al. [194]. Note that harmonic inpainting in 1D is linear interpolation, while biharmonic inpainting is cubic interpolation.

In either case, it is clear that in the 2D setting performing a projection (i.e. tonal optimisation) for each candidate pixel can prove to be prohibitively expensive. Moreover, it is not directly clear how the linear feature formulation can be brought to a form such as (if this is feasible at all):

$$\boldsymbol{u} = \boldsymbol{D}\boldsymbol{C}\boldsymbol{a} + \boldsymbol{W}_{\text{ker}} \boldsymbol{\mu}. \tag{4.38}$$

Table 4.3: Runtime scaling with resolution of our spatial optimisation for FEM harmonic inpainting with $n = 10$ densification iterations and a density of 4%.

image size	64×64	128×128	256×256	512×512	1024×1024
spatial opt.	0.01s	0.03s	0.11s	0.49s	2.07s

An idea would be to consider $\boldsymbol{D} = (\boldsymbol{FL})^+$ (where $\boldsymbol{M}^+$ is the Moore-Penrose inverse), which results in a synthesis and not an analysis approach, but we do not pursue this further here, as it also does not generalise to nonlinear inpainting operators and nonlinear features.

4.2.8 Experiments: FEM Harmonic Inpainting

We illustrate the almost linear runtime-to-resolution scaling of our densification approach applied to the FEM harmonic inpainting in Table 4.3. The complexity of the spatial optimisation is $\mathcal{O}(nq\sqrt{\kappa})$, where n is the number of iterations, κ is the condition number of the matrix, q is the number of non-zero entries in the matrix (at most 6× the number of vertices of the FEM mesh).

We also compare our densification to a standard spatial optimisation approach from the literature – probabilistic sparsification (PS) [156] – which we outperform both qualitatively and runtime-wise. Our densification allows for a trade-off between runtime and quality by increasing the number of densification iterations. For details see Table 4.4.

In terms of runtime, a probabilistic sparsification of a 256×256 image with the parameters from [110] on a Ryzen 4800H CPU takes about 10 minutes, while our FEM densification needs only 0.3 seconds. This shows that our method is 1800 times faster. This factor grows rapidly with the image size: For 512×512 images, it is already 10,000 (4 hours versus 1.3 seconds). And do note that we can qualitatively outperform the PS approach by up to 50% (see the column *walter* in Table 4.4).

Table 4.4: MSE comparisons at 4% density for different numbers of densification iterations n. We outperform prior approaches (PS) qualitatively and runtime-wise.

Method	*trui*		*walter*		*peppers*	
	no TO	with TO	no TO	with TO	no TO	with TO
PS ($q = 10^{-6}$)	66.11	36.04	32.96	19.24	44.85	28.58
Ours ($n = 10$)	44.62	30.07	19.09	12.62	43.20	29.83
Ours ($n = 30$)	40.58	28.21	16.35	**11.09**	38.37	**28.11**
Ours ($n = 100$)	**37.60**	**26.62**	**15.92**	11.21	**36.68**	28.85

4.2.9 Experiments: Feature Inpainting

We also carry out spatial optimisation experiments for our feature inpainting framework. The experimental setup is described in Section 4.2.9.1. We provide visual and qualitative experiments in Section 4.2.9.2 for harmonic inpainting, and in Section 4.2.9.3 for EED inpainting. We briefly mention the setting of nonlinear features in Section 4.2.9.4, demonstrating that the framework is also valid in the nonlinear case, even if the chosen nonlinear feature seems to be suboptimal.

4.2.9.1 Experimental Setup

Image Test Set. We conduct experiments on twelve 512 × 512 natural images that exhibit a wide range of frequencies and structural patterns, see Fig. 2.6 and Fig. 2.7 for a visualisation of eight of those images. For each image, we perform 30 densification iterations, using different sampling densities tailored to each case. The densities are chosen to ensure that the error remains noticeable, allowing for a meaningful visual comparison between inpaintings with different feature sets. The only exception is the nonlinear feature case, where we use only 10 densification iterations as each of those is a 100 times more expensive due to the cost of the nonlinear-constrained feature inpainting. The inpaintings and features are as described in Section 2.5.1. The only difference is that for images that contain higher frequencies such as *shed, raindeer, madeira,* and *crab*, we use a lower pre-smoothing parameter $\sigma = 0.1$ in EED, which results in better quality.

Inpaintings with Linear Constraints. For the linear systems arising in linearly constrained harmonic and EED inpainting we apply the modified conjugate residual solver [52] to (2.16) and (2.23). We use a stopping criterion based on the residual $\|\boldsymbol{r}\|_2 \leq 10^{-8}\|\tilde{\boldsymbol{r}}_0\|_2$ where

$$\|\tilde{\boldsymbol{r}}_0\|_2 = \|(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{L}(\boldsymbol{u}^k)\boldsymbol{P}\boldsymbol{f}\|_2 \tag{4.39}$$

is the residual norm corresponding to an initial guess that satisfies the constraints. The matrix-vector products with the projection $\boldsymbol{P} = \boldsymbol{A}^+\boldsymbol{A}$ are computed by using the conjugate gradient for the normal equations (CGNR) [206], with a stopping criterion of 10^{-12} and an initial iterate zero – this approximates the Moore-Penrose pseudoinverse solution [101].

Inpaintings with Nonlinear Constraints. In the setting of nonlinear equality constraints we use the SQP approach proposed in Section 2.4. As an initial guess to the SQP iteration we take the solution from harmonic inpainting with only the linear equality constraints being active. In each SQP step we use a Steihaug-CGNR solver for the computation of the quasi-normal and tangential directions. We have found the latter to be the most robust for SQP, e.g., compared to other solvers such as the modified conjugate residual solver. The stopping criteria are based on the relative residual of the normal equations $\|\boldsymbol{M}\boldsymbol{r}\| \leq 10^{-8}\|\boldsymbol{M}\boldsymbol{r}_0\|$, where $\boldsymbol{M}$ is the system matrix in (2.36) for the tangential step, and for the quasi-normal step we have $\boldsymbol{M} = \boldsymbol{J}_{\boldsymbol{b}}^\top(\boldsymbol{u}^k)$.

q	*boats* 7%	*elpaso* 3%	*shed* 15%	*generator* 3%	*raindeer* 3%	*quai* 3%
1	144.71	76.03	110.27	175.66	57.21	67.22
3	126.22	62.88	91.82	164.29	55.81	62.47
4	66.53	34.76	80.20	100.52	35.47	41.46
5	**65.26**	**33.85**	**79.41**	**99.48**	**34.55**	**39.59**
q	*mirror 3%*	*madeira 8%*	*garafia 7%*	*flowers 3%*	*crab 8%*	*windmill 5%*
1	127.39	75.09	134.43	195.40	332.53	184.91
3	120.32	66.88	122.16	192.56	292.70	161.65
4	44.98	59.08	70.00	87.46	240.72	81.20
5	**43.01**	**58.53**	**68.16**	**79.17**	**238.12**	**78.94**

Table 4.5: MSEs for harmonic inpainting with an increasing number of feature families.

4.2.9.2 Harmonic Inpainting with Linear Features

We present a series of experiments demonstrating the robustness of our densification approach and its significant improvement over previous state-of-the-art for sparse harmonic inpainting.

Mean Squared Error. Table 4.5 lists the MSEs for all twelve images for an increasing set of feature types $\{\boldsymbol{\phi}_k\}_{k=1}^{q}$ while using the exact same total mask density. We are able to decrease the MSE to be one third for the image *mirror*, and often we are able to halve the MSEs. We note that the most beneficial feature is $\boldsymbol{\phi}_4$ – the 3×3 binomial kernel. As expected, $\boldsymbol{\phi}_5$ – the 5×5 binomial kernel – has little effect once $\boldsymbol{\phi}_4$ is present, since it's just an approximation of Gaussian convolution with a larger standard deviation than for $\boldsymbol{\phi}_4$. The derivative features $\boldsymbol{\phi}_2 \approx \partial_x$, $\boldsymbol{\phi}_3 \approx \partial_y$ have a more subtle effect than $\boldsymbol{\phi}_4$ on natural images – they are much better suited to cartoon-like or piecewise-constant images.

Inpaintings and Masks. Inpaintings along with masks for an increasing variety of considered features are presented in Figure 4.6 and Figure 4.7. The figures demonstrate the effect of increasing the number of features while keeping the same total mask density – the mean squared error (MSE) generally decreases in a monotone way. This testifies to the robustness of our optimisation and also illustrates the viability of the chosen features.

Improved Densification. We also improve upon the results from our previous work [120]. There the five features' MSE for *elpaso* and *windmill* at 5% were respectively 23.25 and 157.30. With our new densification we are able to achieve MSEs of 15.27 and 78.95. This is mainly due to the improvements of our densification algorithm, and partially due to replacing the 2×2 and 16×16 averages with the 3×3 and 5×5 binomial kernels.

q	*boats* 7%	*elpaso* 3%	*shed* 15%	*generator* 3%	*raindeer* 3%	*quai* 3%
1	53.60	25.51	109.65	131.34	59.78	52.33
3	51.80	24.07	93.64	128.23	58.17	51.01
4	35.38	17.98	81.24	97.00	35.15	40.21
5	**33.07**	**17.04**	**79.02**	**93.19**	**33.17**	**37.52**

q	*mirror 3%*	*madeira 8%*	*garafia 7%*	*flowers 3%*	*crab 8%*	*windmill 5%*
1	31.94	75.92	67.26	112.15	328.00	91.00
3	30.42	69.49	67.09	111.95	297.27	85.92
4	19.87	58.88	44.22	74.29	241.04	61.13
5	**19.11**	**57.99**	**42.26**	**67.84**	**236.42**	**56.35**

Table 4.6: MSEs for EED inpainting with an increasing number of feature families.

4.2.9.3 EED Inpainting with Linear Features

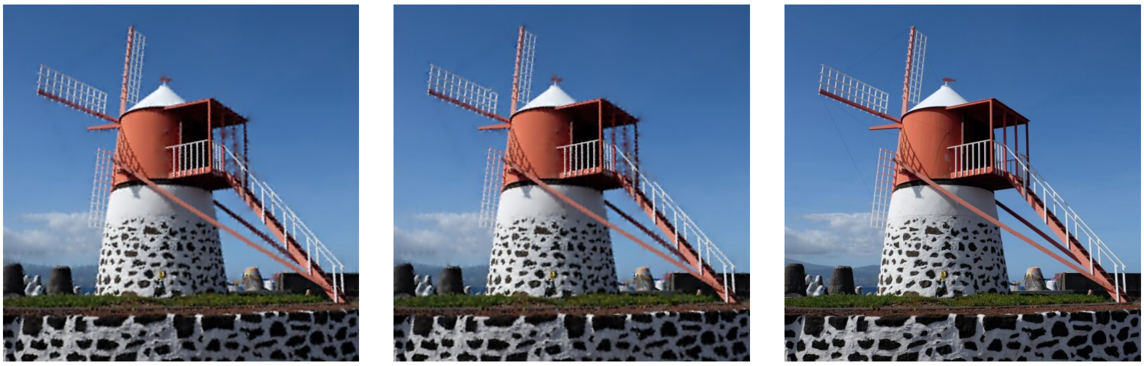
We use edge-enhancing diffusion (EED) for nonlinear inpainting because it has demonstrated very strong performance for sparse image inpainting with low- to mid-frequency content [89]. By using a larger set of features, we are able to improve upon this already strong approach.

Mean Squared Error. Table 4.6 shows the mean squared errors (MSEs) for all twelve images using an increasing set of feature types $\{\boldsymbol{\phi}_k\}_{k=1}^q$, while keeping the total mask density constant. We achieve up to 40% reduction in MSE. Similar to the harmonic inpainting case, the most impactful feature – aside from the point interpolation – is the binomial kernel feature $\boldsymbol{\phi}_4$. In contrast, the derivative features $\boldsymbol{\phi}_2$ and $\boldsymbol{\phi}_3$ have only a minor effect. This is because EED can inherently reconstruct edges from point values alone, making gradient-like features somewhat redundant. This highlights an important principle in feature selection: **features should be chosen to complement the specific weaknesses of the inpainting operator**. In the harmonic inpainting case the 3×3 and 5×5 stencils greatly help with suppressing the logarithmic singularities, while the derivative features help to bring some directional edge data which harmonic inpainting lacks due to its isotropic behaviour. The latter is not the case for EED which can benefit from its strong anisotropy.

Inpaintings and Masks. Figure 4.6 and Figure 4.7 show inpainted images and their corresponding masks for an increasing number of feature types. These examples illustrate the effect of adding more features while keeping the total mask density fixed—resulting in a generally monotonic decrease in mean squared error (MSE). The same images are chosen as in the harmonic inpainting case in order to facilitate comparisons.

4.2.9.4 Inpainting with Nonlinear Features

In the context of nonlinear features, we compare two setups: harmonic inpainting using the first four linear features $\{\boldsymbol{\phi}_k\}_{k=1}^4$, and harmonic inpainting using those same features



lin. constr., MSE: 110.19 nonlin. constr., MSE: 108.24 original

FIGURE 4.5: *windmill* with 10 densification iterations: four linear features (left) vs four linear features and the nonlinear weighted variance feature (middle).

plus the weighted variance feature, i.e., $\{\boldsymbol{\phi}_k\}_{k=1}^4 \cup \{\tilde{\boldsymbol{\phi}}_4\}$. An illustrative example is shown in Fig. 4.5, demonstrating the feasibility of incorporating nonlinear feature constraints. However, the weighted variance feature does not produce significant improvements to justify the added computational cost. Identifying more effective nonlinear features remains an open direction for future research.

4.2.10 Conclusion

We presented an efficient approach for spatial optimisation of both linear and nonlinear inpainting. The approach was general, in the sense that it also applies to our feature inpainting framework. The significant quality gains achieved simply by incorporating additional features emphasise that improving the modelling of the reconstruction process can be as impactful – if not more so – than optimising the data selection itself.

Our experiments with edge-enhancing diffusion (EED) inpainting revealed that derivative features contribute little to its reconstruction quality. This is consistent with the fact that EED already excels at edge completion. The takeaway is that features should be chosen to complement the specific weaknesses of the inpainting operator, targeting structures it cannot reconstruct effectively on its own.

Promising directions for future work include extending the framework to handle inequality constraints, developing gradient-based data selection strategies, exploring machine learning techniques, and identifying nonlinear features that offer a more favourable trade-off between reconstruction quality and computational cost.

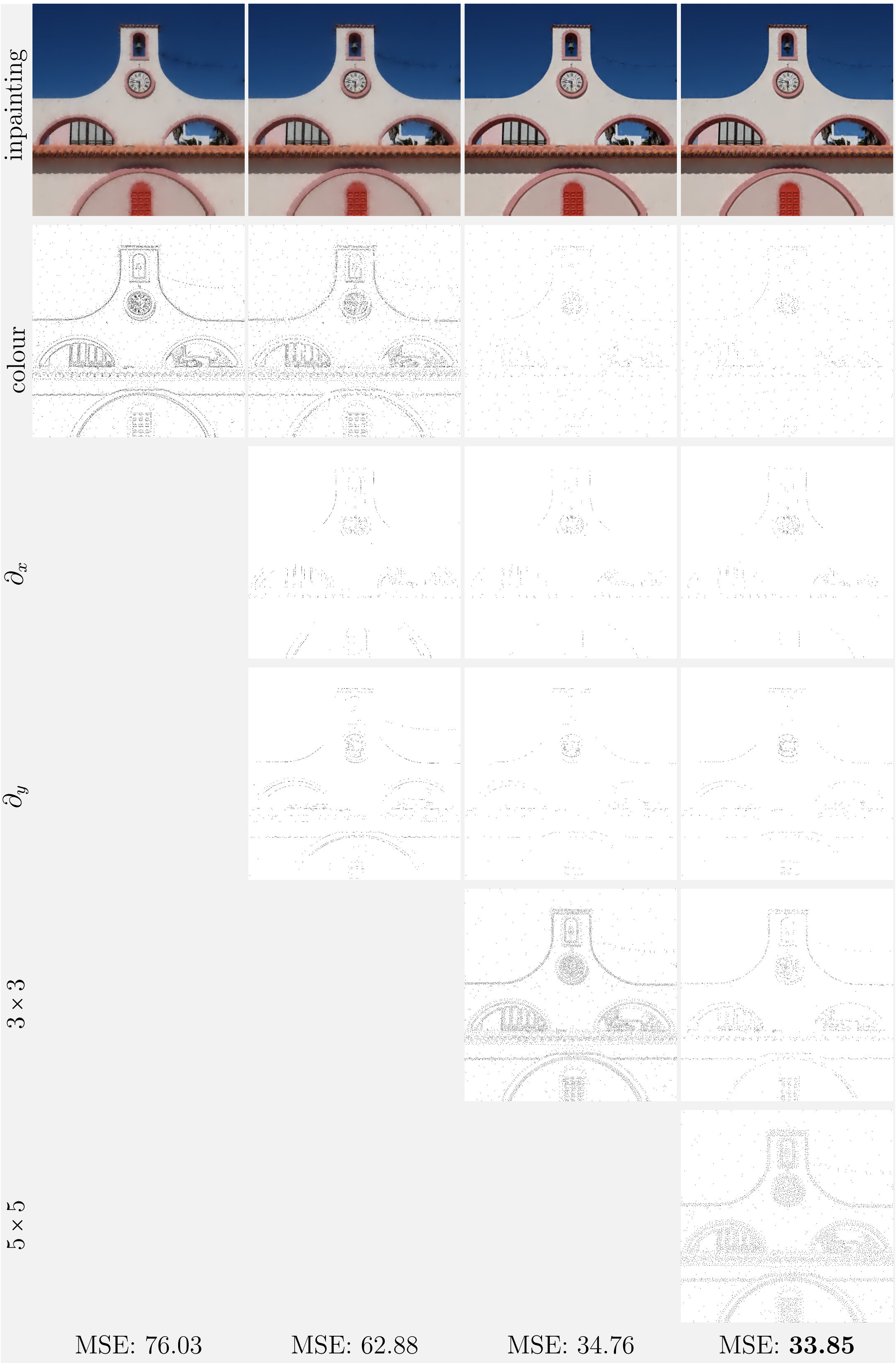


FIGURE 4.6: Harmonic inpainting results for a combined total mask density of 3% for *elpaso* produced with 30 densification iterations.

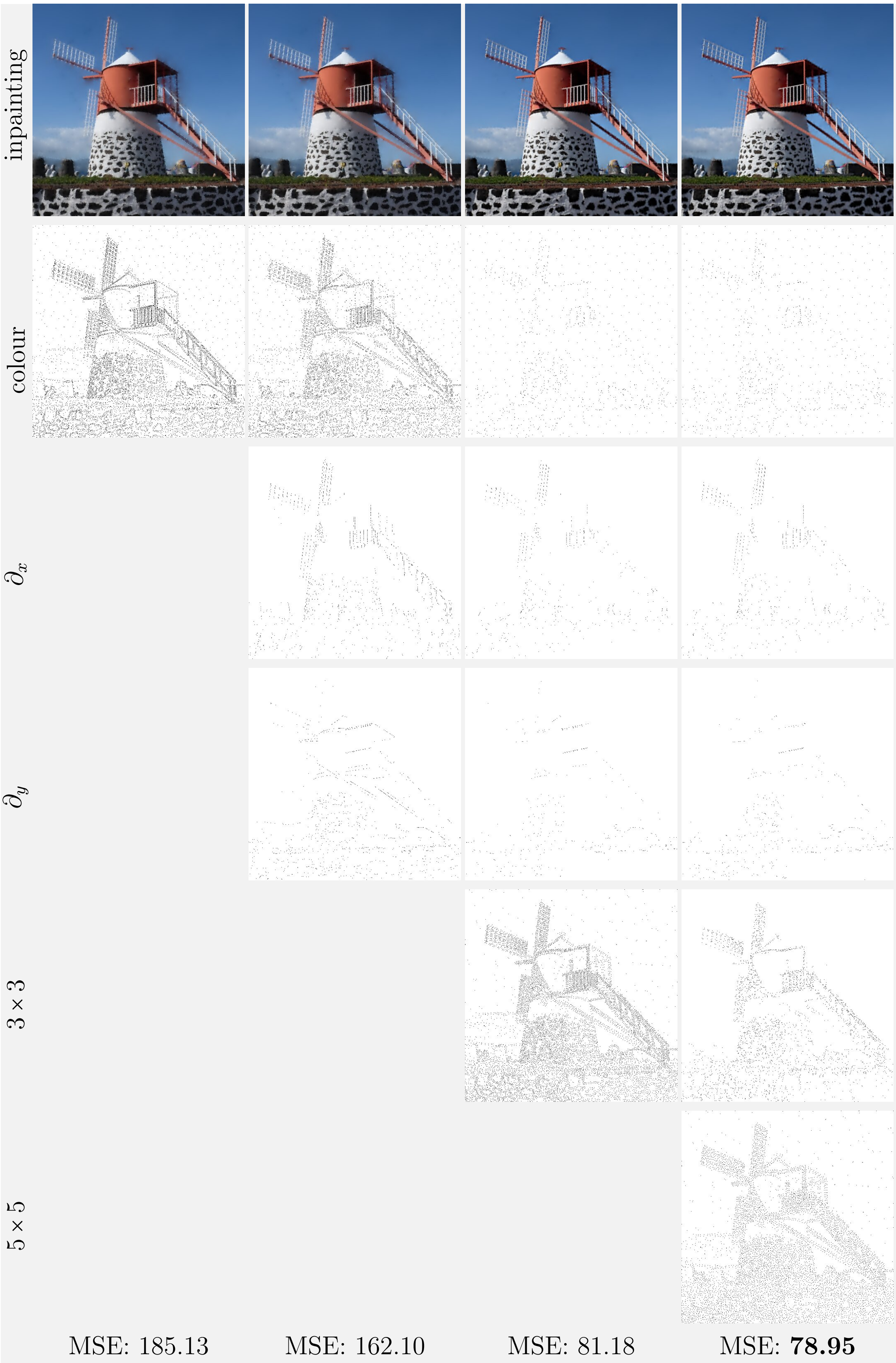


FIGURE 4.7: Harmonic inpainting results for a combined total mask density of 5% for *windmill* produced with 30 densification iterations.

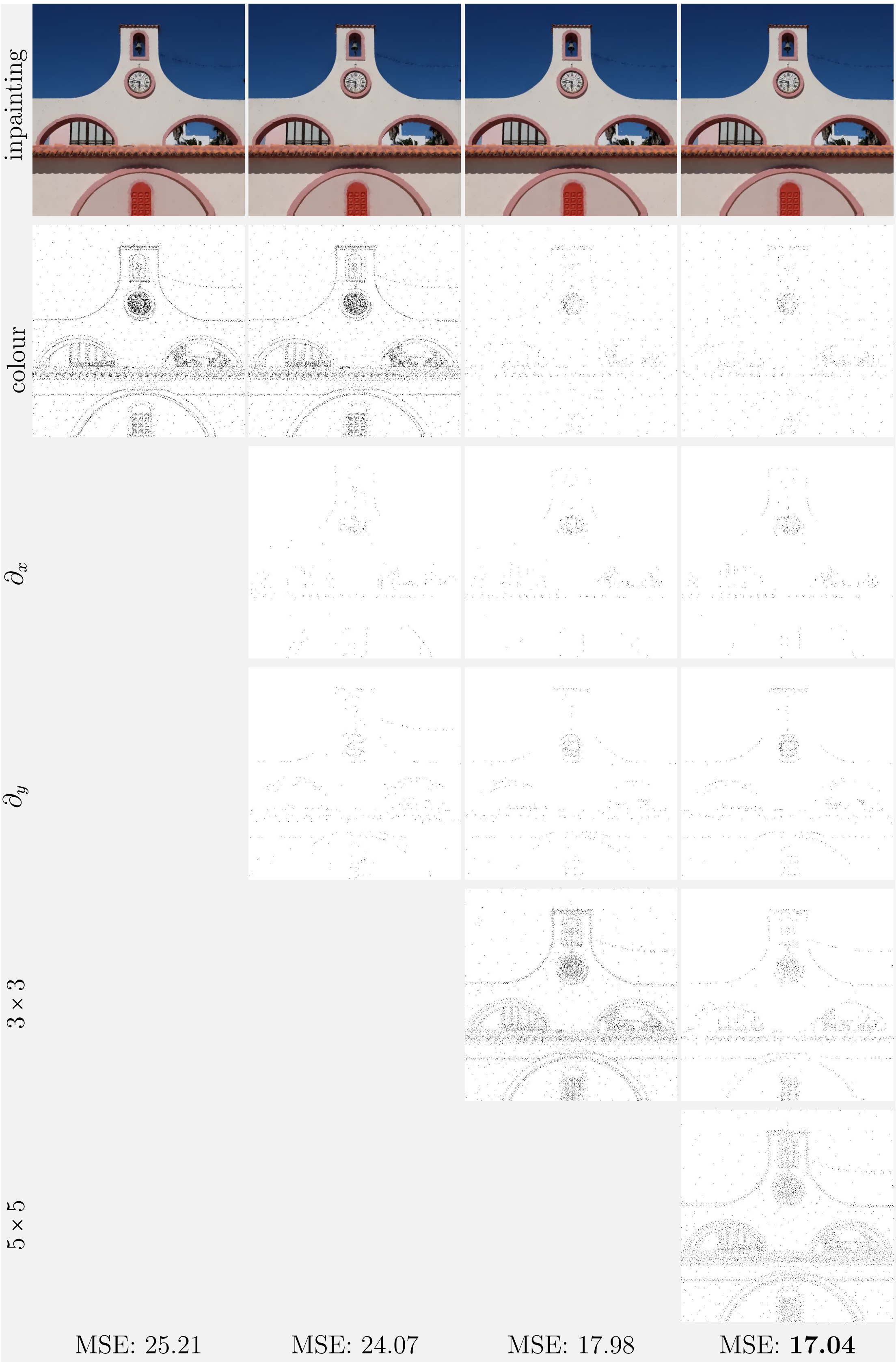


FIGURE 4.8: EED inpainting results for a combined total mask density of 3% for *elpaso* produced with 30 densification iterations

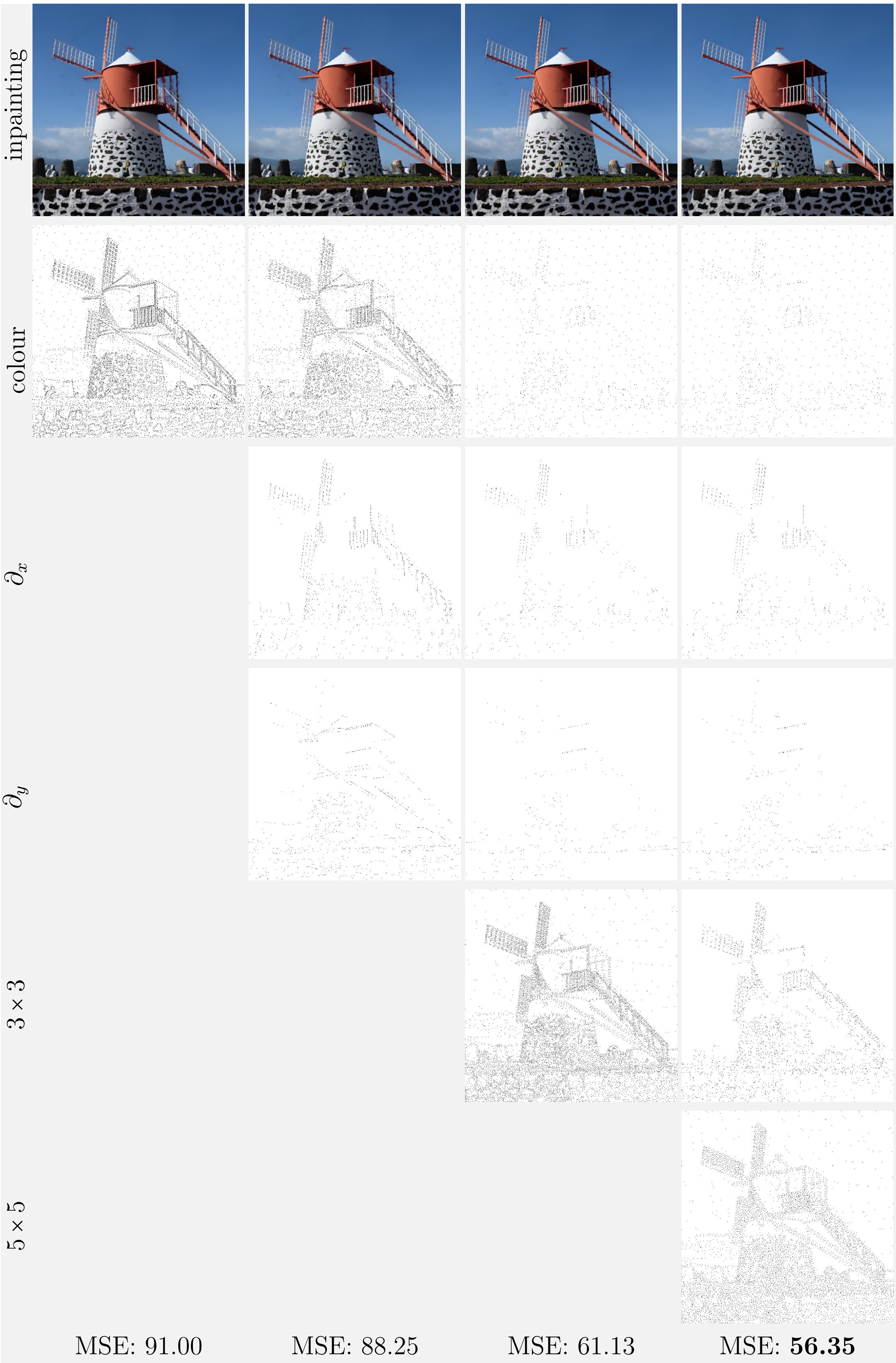


FIGURE 4.9: EED inpainting results for a combined total mask density of 5% for *windmill* produced with 30 densification iterations.

Chapter 5
Denoising by Inpainting

5.1 Introduction

Investigating connections between different fields in image analysis has often been rewarded with deep structural insights. Consider for example the link between variational image inpainting [30, 78, 96, 163, 212] and optic flow computation [114, 173, 237] via the concept of the *filling-in effect.* This effect is due to the smoothness term (regulariser) of the models, which inserts information at locations where the data term is absent or small in magnitude. The gradient flow for minimising the variational energy functional leads to partial differential equations (PDEs) with a diffusion term.

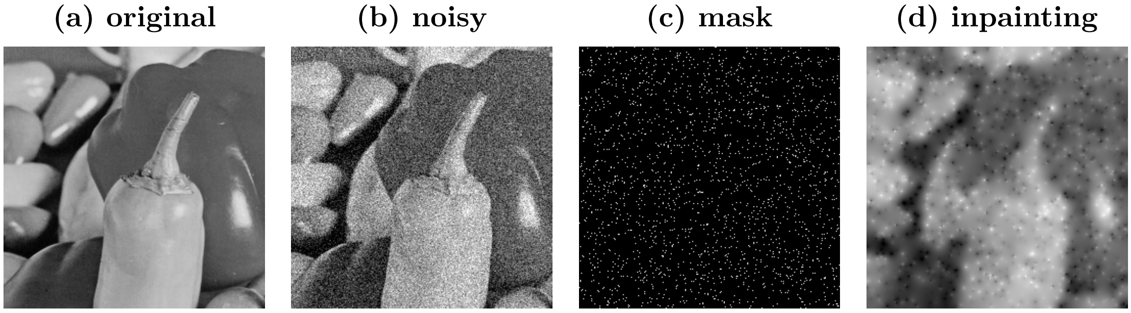
While the filling-in effect has an obvious benefit for image inpainting, it can also lead to more powerful optic flow methods. It produces a dense flow field from the sparse information of the data term. Surprisingly, the parts of the flow field that are filled in by the diffusion-like regularisation terms are usually the ones with the highest confidence [42].

Figure 5.1 shows a similar but hitherto hardly studied effect when performing *sparse inpainting* on noisy data. There the known data – the so-called *mask* – is a scattered set of pixels. The noisy mask pixels remain unchanged during the process, while the unknown areas in between are interpolated smoothly by averaging information from the noisy pixels. We thus again have a scenario, where *the filled-in data are more reliable than the known data.* In the present manuscript we study how far this idea can lead us.

5.1.1 Our Contribution

The goal of our work is to shed some light on the connections between PDE-based inpainting and denoising, two tasks which have coexisted for a long time, while their links have hardly been studied so far. We bridge this gap by a detailed investigation of the unconventional idea of denoising by inpainting. To facilitate a rigorous mathematical analysis, we focus on homogeneous diffusion. As will be explained below, it constitutes the most transparent and most foundational setting in both worlds.

The present chapter builds upon the conference publication [2], in which the basic denoising by inpainting framework is established, and on our journal extension [88]. This



peppers $\sigma_n = 30$ 3% density HD inpainting

Figure 5.1: Homogeneous diffusion (HD) inpainting on the test image *peppers* (256×256 pixels, image range $[0, 255]$) with additive Gaussian noise of standard deviation $\sigma_n = 30$ that we do not clip. The mask pixels are randomly selected. Note that the inpainted pixels are more reliable, since they average noisy information from the neighbourhood. The visual difference is also reflected by the mean squared error (MSE): The MSE of the noisy image in (b) is 904. Since the mask pixels are chosen randomly and are not changed by the inpainting, the MSE at mask pixel locations in (d) is still approximately 900. However, the total image MSE in (d) is only 475.

framework reconstructs a denoised version of an image by averaging the results of multiple inpaintings obtained from distinct masks. Furthermore, two concrete implementations of this framework are proposed in [2]: The first uses shifted regular masks and allows to establish a relation between denoising by inpainting and classical diffusion filtering in 1D, while the second uses probabilistic densification to adapt the masks to the image structures and enables an edge-preserving denoising behaviour.

We extend the aforementioned results by a much broader study of the framework in [2], providing a fundamental understanding of the connections between PDE-based image inpainting and denoising. Since denoising methods can also be used as plug-and-play priors in algorithms for solving inverse problems [147, 203, 234], our relations between inpainting and denoising approaches may have an even broader application spectrum. Compared to [2], we introduce the following additional contributions:

- We show that the heuristically motivated DbI framework from [2] can be seen as a representative of a general probabilistic framework, for which we derive a sound theory. We argue that the denoising result obtained with such framework is an approximation of a minimum mean squared error (MMSE) estimate.
- We provide convergence estimates for the framework and propose a deterministic sampling approach to boost the convergence.
- We prove a general relation between the mask density of regular masks in the DbI framework and the diffusion time of homogeneous diffusion filtering in 1D. We also propose an empirical generalisation of this result to 2D for uniform random masks.
- We integrate a step that optimises the grey values at the selected mask pixels (tonal optimisation) into the DbI framework. We investigate its effect on the

MMSE estimate and perform experiments which confirm that tonal optimisation can improve the denoising performance of DbI in practice.

- We show that the different spatial optimisation approaches in the DbI framework correspond to specific posterior distributions. We compare two such strategies (the one presented in [2] and a novel one) in terms of quality and provide the formulations for the respective probability distributions. Our experiments demonstrate that this data optimisation leads to an edge-preserving denoising behaviour.
- We replace homogeneous diffusion inpainting in the DbI framework by biharmonic inpainting and show that it is unable to improve denoising results. This confirms one of our key insights: The hitherto hardly practised data optimisation can be as powerful as widely used operator optimisations.

Why Homogeneous Diffusion ? Our decision to focus on homogeneous diffusion is based on several reasons:

- For denoising and image simplification, one should keep in mind that homogeneous diffusion filtering is equivalent to Gaussian convolution. The Gaussian is the only convolution kernel that is separable and rotation invariant. The diffusion evolution generates a Gaussian scale-space representation [115, 152, 219], which is one of the most widely-used scale-spaces and forms the basis of highly successful interest point detectors such as SIFT [153] and its numerous variants.
- In inpainting applications, homogeneous diffusion is particularly popular in inpainting-based compression [89], where one stores only a sparse subset of all pixels and reconstructs the image in the decoding phase by inpainting. By optimising the stored data, homogeneous diffusion can achieve surprisingly faithful reconstructions [156]. Moreover, its simplicity allows a detailed theoretical analysis [28], it frees the user from specifying parameters, and one can achieve real-time performance on current PC hardware even for large images [126].
- Last but not least, there exist already well-understood connections between diffusion processes for denoising and other approaches, such as variational regularisation methods [176, 207] and wavelets [74, 240], but also deep neural network architectures [10, 205]. Thus, establishing also connections to inpainting ideas gives more comprehensive insights into various paradigms beyond diffusion-based denoising.

This discussion also implies that **it is not the goal of the present chapter to design novel approaches that outperform the most recent state-of-the-art approaches for denoising or inpainting.** This is reserved for future research that may benefit from the foundational insights in our manuscript.

5.1.2 Outline

We review related work in Section 5.1.3. In Section 5.2 we briefly introduce the basic idea behind diffusion filtering and its application to image denoising and image inpainting.

In Section 5.3 we present the framework for denoising by inpainting from [2] and show that it can be interpreted as a Monte Carlo approach for approximating an MMSE estimate. We additionally provide convergence results, and suggest a method to boost the convergence by employing low-discrepancy sequences. In Section 5.4 we relate denoising by inpainting with non-adaptive masks to classical diffusion filtering. In Section 5.5 we present strategies for adaptively selecting the mask pixels in the DbI framework, which leads to space-variant denoising behaviour. Our experiments and results are presented in Section 5.6, and we conclude the chapter in Section 5.7.

5.1.3 Related Work

Since we consider image inpainting as well as image denoising, we give an overview of some relevant methods from both fields and relate them to our work.

PDE-based Denoising and Inpainting. We borrow several ideas from sparse PDE-based inpainting methods [89]. We mostly restrict ourselves to homogeneous diffusion inpainting [46], which can be implemented very efficiently [14, 58, 113, 125, 126, 155], and – in spite of its simplicity – can produce convincing results for suitably chosen data [28, 35, 39, 57, 109, 111, 156, 178]. Especially on piecewise constant images, such as cartoon images, depth maps or flow fields, homogeneous diffusion inpainting in conjunction with edge or segment information performs very well [46, 90, 112, 118, 119, 149, 155]. This even allows some of these methods [118, 119] to outperform HEVC [222] on such data. Nonlinear diffusion inpainting methods, e.g., edge-enhancing diffusion (EED) inpainting [89, 238], can improve reconstruction quality for sparse inpainting, enabling lossy image codecs [89, 121, 209] competitive to JPEG [185] and JPEG2000 [225]. On the other hand, such methods are more complex due to their nonlinearity. This complexity also carries over to the data optimisation process. Higher-order inpainting operators can also be used for sparse inpainting [44, 57, 89, 209, 224], but can be more sensitive to noise. The quality of PDE-based sparse inpainting approaches strongly depends on the stored data, and in our denoising by inpainting framework we incorporate ideas from spatial optimisation [28, 35, 57, 58, 109, 111, 155, 156, 178] and tonal optimisation [57, 58, 111, 156, 187]. To interpret the filtering results of the denoising by inpainting framework, we compare to classical diffusion-based image denoising methods. Aside from the simple homogeneous diffusion [115], we also consider methods that adapt the diffusion operator to the given image, namely linear space-variant diffusion [87] and nonlinear diffusion [186]. We choose these methods because they are closest conceptually so we expect them to provide useful insights.

Patch-based Denoising and Inpainting. Patch- or exemplar-based methods are another class of inpainting methods, and work especially well with textured data. The idea is to copy similar patches from known to unknown regions. Efros and Leung have proposed the first exemplar-based inpainting method [78], but many versions have been developed since then (e.g., [4, 17, 21, 63]), including the method of Facciolo et al. for sparse inpainting [83]. Inpainting approaches combining PDE-based and patch-based methods have also been presented [31, 189]. Inspired by the method of Efros and Leung [78], a patch-based denoising method called NL-means [43] has been proposed. It denoises an image based on a nonlocal weighted averaging of similar image patches. Other algorithms such as the famous BM3D algorithm [64] are also based on the filtering of image patches. These observations further substantiate the ties between denoising and inpainting. The NL-means method can even be interpreted as a case of a denoising by inpainting approach, although it does not use the inpainting ideas as directly as we do. Of course, a direct application of patch-based inpainting techniques would lead to the copying of erroneous noisy data, and not to a denoising effect.

Sparse Signal Approximation. A popular approach in the field of image denoising relies on the idea that signals (and images) can be represented as a linear combination of a smaller number of basis signals – so-called atoms – that are selected from a dictionary [81]. Such a dictionary might for example consist of the basis vectors of a suitable transform, that makes the signal representation sparse (e.g., a wavelet transform [157] or a discrete cosine transform (DCT) [6]). The task is to find the atoms that best represent the given signal [56, 75, 158]. To fill in missing information in images, several authors also consider sparse representations in some transform domain such as the DCT [97] or the shearlet domain [134]. This shows another bridge between the two tasks of denoising and inpainting. Hoffmann et al. [113] relate linear PDE-based inpainting methods to concepts from sparse signal approximation. They solve the inpainting problem with the help of discrete Green's functions [29, 60], which can be interpreted as atoms in a dictionary. This allows for a sparse representation of the inpainting solution. Kalmoun et al. [125] follow a similar approach by solving homogeneous diffusion inpainting with the charge simulation method [132, 141]. An application of homogeneous diffusion inpainting with Green's functions is the video codec by Andris et al. [14]. We justify certain design choices within the DbI framework with results from this field. Notably, homogeneous diffusion inpainting is based on the idea that the Laplacian of the reconstructed image is mostly sparse. On the other hand, the DbI framework combines multiple noisy sparse representations in order to get a denoised but non-sparse representation. The latter can be studied rigorously from a Bayesian denoising perspective, which is why we discuss this next.

Bayesian Denoising. The study of denoising has also been carried out from a probabilistic perspective. Here, the assumption is that some prior information regarding the noise distribution and/or the image distribution is available. This can be incorporated in a denoising framework through Bayes' rule, such that the final denoised result is conditioned on this information about the distributions. The latter provides a correspondence between classical denoising variational methods and specific Bayesian priors [82, 99, 148]. The standard approach is to employ statistical inference approaches, such as maximum likelihood (ML) estimation, maximum a posteriori (MAP) estimation, or minimum mean squared error (MMSE) estimation. Both the MAP and MMSE approach rely on a posteriori density, and as such they require a model of the distribution of considered classes of images. One of the first such models uses a Gibbs distribution for the prior [92]. Subsequently, a number of works have built upon this idea. The most relevant to our setting is that by Larsson and Selen [145], which studies MMSE estimation in the context of sparse vector representations. Our sparse inpaintings can be interpreted as such sparse vector representations. Moreover, in the current work we show that the averaging performed in [2] is in fact a Monte Carlo approach to approximate an MMSE estimate.

Cross-Validation. We also see the work of Craven and Wahba [62] on (generalised) cross-validation as conceptually related to parts of our work. Cross-validation can be used to optimise parameters in denoising models [62, 145, 248]. It removes data points from given noisy observations and judges the quality of a parameter selection in terms of the model's capability to reconstruct the data at these locations. Related ideas are also pursued in [50]. Probabilistic densification [112] and sparsification [156], two concepts from spatial optimisation that we consider in our framework, also use the error of the inpainted reconstruction at left out locations – in our case also on noisy data. Yet, both applications differ, as the goal of the latter methods is to construct an inpainting mask and not to optimise model parameters.

Neural Denoising and Inpainting. In recent years, many very powerful methods for inpainting and denoising have been proposed that rely on neural networks. They are, however, not a topic of our paper, since we aim at gaining structural insights into the connections between inpainting and denoising. Such results on classical approaches are still relevant in the learning era [82]. They may serve as foundations for deep learning-based methods, and model- and learning-based approaches may be fused to obtain powerful and transparent algorithms. It is our hope that in the long run, our insights can also be beneficial to neural approaches.

5.2 Basics of Diffusion Filtering

In its original context of physics, diffusion is a process that equilibrates particle concentrations. When working with images, we interpret the grey values as particle concentrations and use diffusion processes as smoothing filters that balance grey value differences. To this end, we define the original greyscale image as a function $f : \Omega \to \mathbb{R}$, with $\Omega \subset \mathbb{R}^2$ being a rectangular image domain. Similarly, $u : \Omega \times [0, \infty) \to \mathbb{R}$ denotes the evolving, filtered image. Then the diffusion evolution is described by the following PDE:

$$\partial_t u(\boldsymbol{x}, t) = \operatorname{div}(g \boldsymbol{\nabla} u(\boldsymbol{x}, t)) \quad \text{for } \boldsymbol{x} \in \Omega,\ t \in (0, \infty). \tag{5.1}$$

Here t denotes time, $\boldsymbol{\nabla} = (\partial_x, \partial_y)^\top$ is the spatial gradient, $\operatorname{div}(\boldsymbol{v}) = \partial_x v_x + \partial_y v_y$ is the spatial divergence, and the scalar diffusivity g determines the local smoothing activity. We discuss different choices of g in Section 5.2.1. Note that g can be extended to a diffusion tensor to introduce anisotropy into the process [236], but since we do not consider such a case in this chapter, we refrain from discussing it here. We equip the PDE with an initial condition at time $t = 0$ and reflecting boundary conditions at the image boundary $\partial\Omega$:

$$u(\boldsymbol{x},0) = f(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \in \Omega, \tag{5.2}$$

$$\partial_{\boldsymbol{n}} u(\boldsymbol{x},t) = 0 \quad \text{for } \boldsymbol{x} \in \partial\Omega,\ t \in (0,\infty), \tag{5.3}$$

where $\boldsymbol{n}$ is the outer normal vector at the image boundary. Solving this initial boundary value problem for u yields a family of filtered images $\{u(\cdot,t)\mid t \geq 0\}$.

5.2.1 Diffusion for Image Denoising

In image denoising the image f is a noisy version of the noise-free ground truth image f_r. In our case we assume zero-mean additive white Gaussian noise, i.e., $f = f_r + n$ with $n \in \mathcal{N}(0,\sigma_n^2)$. Diffusion processes are good candidates for image denoising tasks thanks to their smoothing properties. Depending on the form of the diffusivity g, different processes are obtained.

5.2.1.1 Homogeneous Diffusion

By setting $g \equiv 1$, (5.1) simplifies to $\partial_t u = \Delta u$, with $\Delta u = \partial_{xx} u + \partial_{yy} u$ being the Laplacian operator. The resulting process is known as *homogeneous diffusion* [115]. Its analytical solution in the unbounded image domain $\mathbb{R}^2$ is given by a convolution of the original image with a Gaussian kernel K_σ with standard deviation $\sigma = \sqrt{2t}$. The resulting images $\{u(\cdot,t)\mid t \geq 0\}$ constitute the so-called Gaussian scale-space [115, 239]. Since g is selected to be constant, the smoothing strength is the same across the entire image. Therefore, not only the noise is reduced, but also semantically important image structures such as edges are smoothed.

5.2.1.2 Linear Space-Variant Diffusion

To overcome the drawbacks of homogeneous diffusion, one can make the process space-variant by selecting a diffusivity function that varies depending on the structure of the *initial* image f [87]. This is called *linear space-variant diffusion.* If edges and other high-gradient features are to be preserved, the diffusivity should be decreasing with increasing gradient magnitude of the image, so that that the smoothing would be reduced at edges. An example for a suitable function is the Charbonnier diffusivity [54]:

$$g(|\nabla f|^2) = \frac{1}{\sqrt{1 + \frac{|\nabla f|^2}{\lambda^2}}}, \tag{5.4}$$

where $|\cdot|$ denotes the Euclidean norm. The contrast parameter $\lambda > 0$ is used to distinguish locations where smoothing should be applied (for $|\nabla f| \ll \lambda$, we get $g_\lambda \to 1$) and locations where it should be reduced (for $|\nabla f| \gg \lambda$, we obtain $g_\lambda \to 0$).

5.2.1.3 Nonlinear Diffusion

Alternatively, one can make the diffusivity function g dependent on the *evolving* image u. This allows to update the locations where smoothing is reduced during the evolution, by choosing them based on the image u, which becomes gradually smoother and less noisy. The resulting process $\partial_t u = \operatorname{div}(g(|\nabla u|^2)\nabla u)$ is *nonlinear* [186]. The feedback mechanism throughout the evolution helps steering the process to achieve better results.

5.2.2 Diffusion for Image Inpainting

Diffusion processes can also be used to fill in missing information in images [46, 51, 238]. Particularly, they allow to reconstruct an image from only a small number of pixels by propagating information from known to unknown areas [89]. The set of known pixels is called the *inpainting mask* and is denoted by $K \subset \Omega$. To recover the image, the information at the unknown locations is computed as the steady state ($t \to \infty$) of a diffusion process, while the values at mask locations are preserved. The parabolic inpainting formulation is obtained by modifying (5.1) and (5.2) accordingly:

$$\begin{aligned} \partial_t u(\boldsymbol{x},t) &= \operatorname{div}(g\nabla u(\boldsymbol{x},t)) && \text{for } \boldsymbol{x} \in \Omega \setminus K,\ t \in (0,\infty), && (5.5)\\ u(\boldsymbol{x},t) &= f(\boldsymbol{x}) && \text{for } \boldsymbol{x} \in K\,,\ t \in [0,\infty), && (5.6)\\ u(\boldsymbol{x},0) &= 0 && \text{for } \boldsymbol{x} \in \Omega \setminus K, && (5.7)\\ \partial_{\boldsymbol{n}} u(\boldsymbol{x},t) &= 0 && \text{for } \boldsymbol{x} \in \partial\Omega,\ t \in (0,\infty). && (5.8) \end{aligned}$$

For $g \equiv 1$, (5.5) is the homogeneous diffusion PDE [115] and we talk about *homogeneous diffusion inpainting* (also called *harmonic inpainting*). We almost exclusively consider homogeneous diffusion inpainting in the remainder of this chapter, so we set $g \equiv 1$ in the following. Instead of computing the steady state of the parabolic diffusion equation, we may solve the corresponding boundary value problem:

$$\begin{aligned} -\Delta u(\boldsymbol{x}) &= 0 && \text{for } \boldsymbol{x} \in \Omega \setminus K, && (5.9)\\ u(\boldsymbol{x}) &= f(\boldsymbol{x}) && \text{for } \boldsymbol{x} \in K, && (5.10)\\ \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0 && \text{for } \boldsymbol{x} \in \partial\Omega. && (5.11) \end{aligned}$$

The problem may be written equivalently using the variational formulation

$$\min_u \int_\Omega |\nabla u(\boldsymbol{x})|^2 \, d\boldsymbol{x}, \text{ such that } u(\boldsymbol{x}) = f(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in K. \tag{5.12}$$

This suggests the interpretation that the inpainting is designed to penalise the gradient magnitude of the reconstruction, i.e., it inherently promotes smoothness. In order to simplify the discretisation of the boundary value problem formulation, we introduce a mask indicator function $c = \mathbb{1}_K$ (we use the term *mask* synonymously for the set K and the function c), that takes the value 1 at points from K and 0 elsewhere. This allows us to combine (5.9) and (5.10) into a single equation

$$\big(c(\boldsymbol{x}) + (1 - c(\boldsymbol{x}))(-\Delta)\big)u(\boldsymbol{x}) = c(\boldsymbol{x})f(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \in \Omega. \tag{5.13}$$

5.2.3 Discrete Homogeneous Diffusion Inpainting

Since we are working with digital images, the above considerations need to be translated to the discrete setting. We therefore discretise the images on a regular pixel grid of size $n_x \times n_y$. Then we write them as vectors of length $N = n_x n_y$ that are obtained by stacking the discrete images column-by-column, e.g., $\boldsymbol{f}, \boldsymbol{u} \in \mathbb{R}^N$. Furthermore, let $\boldsymbol{L} \in \mathbb{R}^{N\times N}$ denote the five-point stencil discretisation matrix of the negated Laplacian $(-\Delta)$ with reflecting boundary conditions $\partial_{\boldsymbol{n}} u(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \in \partial\Omega$. Additionally, let $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$ be the diagonal matrix with the mask vector $\boldsymbol{c} \in \{0,1\}^N$ discretising c, and let $\boldsymbol{I}$ be the $N \times N$ identity matrix. Then the discrete version of (5.13) can be formulated as the linear system of equations:

$$(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})\, \boldsymbol{u} = \boldsymbol{C}\boldsymbol{f}, \tag{5.14}$$

and the reconstruction can be written explicitly as

$$\boldsymbol{u} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f}) = (\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L})^{-1}\, \boldsymbol{C}\boldsymbol{f}. \tag{5.15}$$

The inverse of the *inpainting matrix* $\boldsymbol{M_c} \coloneqq \boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\boldsymbol{L}$ exists as long as $\boldsymbol{C} \neq \boldsymbol{0}$ [155]. To deal with the case $\boldsymbol{C} = \boldsymbol{0}$ we define $\boldsymbol{r}(\boldsymbol{0}, \boldsymbol{f}) \coloneqq \frac{1}{N}\boldsymbol{1}^\top \boldsymbol{f}$, i.e., we take the average. If we want to approximate the image $\boldsymbol{f}$ instead of interpolating it over $\boldsymbol{C}$, we can replace $\boldsymbol{C}\boldsymbol{f}$ with $\boldsymbol{C}\boldsymbol{g}$, where

$$\boldsymbol{g} \in \underset{\boldsymbol{h}\,:\,\boldsymbol{h}|_{\bar{\boldsymbol{c}}}=\boldsymbol{0}}{\arg\min}\, \|\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{h}) - \boldsymbol{f}\|_2^2. \tag{5.16}$$

Here $\boldsymbol{h}|_{\boldsymbol{c}}$ is the restriction of $\boldsymbol{h}$ to $\boldsymbol{c}$ and $\boldsymbol{h}|_{\bar{\boldsymbol{c}}}$ is the restriction of $\boldsymbol{h}$ to the complement $\bar{\boldsymbol{c}} = \boldsymbol{1} - \boldsymbol{c}$. The optimisation is thus only over $\boldsymbol{h}|_{\boldsymbol{c}}$ since the remainder of the values are irrelevant for the inpainting result, so we set them to zero. The least squares problem is known as the *tonal optimisation* problem and we discuss its implications for the current work in Section 5.3.1.1. Additionally, we observe that the reconstruction is linear in $\boldsymbol{g}$. This motivates us to write it as a linear combination of basis vectors with weights given by $\boldsymbol{g}|_{\boldsymbol{c}}$. Let $\boldsymbol{B_c} \coloneqq (\boldsymbol{M_c}^{-1})|_{\boldsymbol{I}\times\boldsymbol{C}}$ be the restriction of $\boldsymbol{M_c}^{-1}$ to the columns corresponding to non-zeros in $\boldsymbol{c}$, and we set $m = \|\boldsymbol{c}\|_0$ to be the number of non-zeros in $\boldsymbol{c}$. By denoting the columns as $\{\boldsymbol{b_c^k}\}_{k=1}^m$, i.e., $\boldsymbol{B_c} = [\boldsymbol{b_c^1} \ldots \boldsymbol{b_c^m}]$, we can write the reconstruction as

$$\boldsymbol{u} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) = \boldsymbol{M_c}^{-1}\boldsymbol{C}\boldsymbol{g} = \boldsymbol{B_c}\,\boldsymbol{g}|_{\boldsymbol{c}} = \sum_{k=1}^{m} (\boldsymbol{g}|_{\boldsymbol{c}})_k\, \boldsymbol{b_c^k}. \tag{5.17}$$

We see that the columns of $\boldsymbol{B_c}$ are the basis vectors induced from $\boldsymbol{r}$ and $\boldsymbol{c}$. They are also termed *inpainting echoes* [77, 156]. We note that inpainting with $\boldsymbol{g}|_{\boldsymbol{c}} = \boldsymbol{f}|_{\boldsymbol{c}}$ constructs the interpolant over $\boldsymbol{c}$ in the space $\operatorname{span}(\boldsymbol{B_c}) \subseteq \mathbb{R}^N$. Since the tonal optimisation solution can be written as $\boldsymbol{g}|_{\boldsymbol{c}} = (\boldsymbol{B_c})^+\boldsymbol{f}$, where $(\boldsymbol{B_c})^+$ is the Moore-Penrose pseudo-inverse, we note that $\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{g}) = \boldsymbol{B_c}(\boldsymbol{B_c})^+\boldsymbol{f}$ is the orthogonal projection of $\boldsymbol{f}$ on the subspace $\operatorname{span}(\boldsymbol{B_c}) \subseteq \mathbb{R}^N$, i.e., the best approximant of $\boldsymbol{f}$ in this space.

5.3 Our Denoising by Inpainting Framework

We now present the basic idea and the framework for denoising by inpainting proposed in the conference paper [2] upon which our published work [88] and this chapter are based. Since the framework inherently links inpainting and denoising, it is well-suited to study connections between the two tasks. As previously mentioned, we use diffusion-based inpainting – specifically homogeneous diffusion inpainting – for image denoising, by only keeping a sparse subset of the noisy input data and by reconstructing the rest. Inpainting on noisy images differs from the classical setting and poses additional challenges. During the inpainting process, grey values at mask locations are not altered. As they might contain errors from the noise, these mask pixels are less trustworthy than inpainted pixels, which combine information from their surrounding mask pixels. While we want to exploit the filling-in effect in unknown areas, this observation implies that a single inpainted image cannot give satisfactory denoising results. Therefore, we compute multiple inpaintings with different masks and obtain the final result by averaging them. This ensures that none of the pixels remain unchanged (unless a pixel is contained in all masks). In the current work, we further mitigate the issue of noisy mask pixels by employing tonal optimisation (see Section 5.3.1.1). If we denote the n different masks by $\{\boldsymbol{c}^\ell\}_{\ell=1}^n$, we can generate the inpaintings $\{\boldsymbol{v}^\ell\}_{\ell=1}^n$ via

$$\boldsymbol{v}^\ell = \boldsymbol{r}(\boldsymbol{c}^\ell, \boldsymbol{f}) = \left(\boldsymbol{C}^\ell + \left(\boldsymbol{I} - \boldsymbol{C}^\ell\right)\boldsymbol{L}\right)^{-1}\boldsymbol{C}^\ell\boldsymbol{f}. \tag{5.18}$$

We obtain the final denoising result $\langle \boldsymbol{u} \rangle_n$ by averaging:

$$\langle \boldsymbol{u} \rangle_n = \frac{1}{n}\sum_{\ell=1}^{n} \boldsymbol{v}^\ell = \frac{1}{n}\sum_{\ell=1}^{n} \boldsymbol{r}(\boldsymbol{c}^\ell, \boldsymbol{f}). \tag{5.19}$$

As we fix the inpainting operator (for a discussion of denoising by biharmonic inpainting see Section 5.6.4), the only freedom in the framework lies in the selection of the different masks. This is in contrast to the common strategy in denoising, where all available data is used and the operator is optimised instead. To study the effects of different data selection strategies, we will borrow several ideas from mask optimisation for image compression. To obtain multiple different masks as our framework requires, we rely on some degree of randomness in the mask generation processes (see Section 5.5). Since we make use of stochastic strategies, we formalise and study DbI from a probabilistic point of view in the following subsection.

5.3.1 Probabilistic Theory

As seen in (5.19), the denoised image is the result of averaging n inpaintings from n different masks, that are generated by some mask optimisation process. In the following, we interpret this from a probabilistic point of view. This allows us to formalise the DbI framework from the conference paper [2] and provides us with tools to study and boost the convergence of our methods in Section 5.3.1.4 and Section 5.3.1.5, respectively. We take the masks $\{\boldsymbol{c}^\ell\}_{\ell=1}^n$ to be independent and identically distributed samples from a predetermined distribution conditioned on $\boldsymbol{f}$, with a conditional probability mass function (PMF) $p(\boldsymbol{c}|\boldsymbol{f})$. Then the estimator $\boldsymbol{u}$ converges to the following conditional expectation for $n \to \infty$:

$$\mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}] = \mathbb{E}\left[\frac{1}{n}\sum_{\ell=1}^{n}\boldsymbol{r}(\boldsymbol{c}^\ell,\boldsymbol{f})\middle|\boldsymbol{f}\right] = \frac{1}{n}\sum_{\ell=1}^{n}\mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})|\boldsymbol{f}] = \sum_{\boldsymbol{c}\in\{0,1\}^N}\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})\,p(\boldsymbol{c}|\boldsymbol{f}). \tag{5.20}$$

The second equality holds because the masks were assumed to be identically distributed, and thus $\mathbb{E}[\boldsymbol{r}(\boldsymbol{c}^\ell,\boldsymbol{f})|\boldsymbol{f}] = \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})|\boldsymbol{f}]$ for any $\boldsymbol{c}$ sampled with the same PMF p. The fourth equality follows from the definition of the conditional mathematical expectation. We note that from this probabilistic point of view, spatial adaptivity is provided through the design of the PMF p. The following proposition shows that the DbI result constitutes a minimum mean squared error (MMSE) estimate. This emphasises its optimality under certain assumptions.

Proposition 1 (DbI as an MMSE Estimate)**.** *The expectation* (5.20) *of the DbI averaging* (5.19) *can be interpreted as an MMSE estimate under prior assumptions on the image and noise distributions, i.e., it solves the minimisation problem*

$$\min_{\boldsymbol{u}\in\mathbb{R}^N}\mathbb{E}[\|\boldsymbol{u}-\boldsymbol{w}\|_2^2|\boldsymbol{f}] = \min_{\boldsymbol{u}\in\mathbb{R}^N}\mathbb{E}[\|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})\|_2^2|\boldsymbol{f}]. \tag{5.21}$$

Proof. We can rewrite the minimisation problem (5.21) as

$$\min_{\boldsymbol{u}\in\mathbb{R}^N}\mathbb{E}[\|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})\|_2^2|\boldsymbol{f}] = \min_{\boldsymbol{u}\in\mathbb{R}^N}\sum_{\boldsymbol{c}\in\{0,1\}^N}\|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})\|_2^2\,p(\boldsymbol{c}|\boldsymbol{f}). \tag{5.22}$$

Taking the derivative w.r.t. $\boldsymbol{u}$ and setting it to zero results in the MMSE estimate

$$\boldsymbol{u}^{\text{MMSE}} = \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})|\boldsymbol{f}] = \sum_{\boldsymbol{c}\in\{0,1\}^N}\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})\,p(\boldsymbol{c}|\boldsymbol{f}). \tag{5.23}$$

By (5.20) this is the same as the expectation $\mathbb{E}[\langle \boldsymbol{u}\rangle_n]$ of the DbI estimator $\langle \boldsymbol{u}\rangle_n$. □

The estimate $\boldsymbol{u}^{\text{MMSE}}$ is close to $\boldsymbol{f}_r$ (and $\langle \boldsymbol{u}\rangle_n$ is close to $\boldsymbol{f}_r$), whenever $\boldsymbol{v} = \boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})$ with $\boldsymbol{c} \sim p(\boldsymbol{c}|\boldsymbol{f})$ provides a good model for the distribution from which $\boldsymbol{f}_r$ is assumed to originate. This formalisation of DbI as an estimator for the MMSE estimate therefore provides an additional justification for the DbI framework as an image denoising approach.

5.3.1.1 MMSE and Tonal Optimisation

The classical DbI formulation (5.19) from [2] employs an *interpolating* inpainting. It is natural to extend the framework to the best *approximating* inpainting, computing the denoised image $\langle \boldsymbol{u}\rangle_n$ as

$$\langle \boldsymbol{u}\rangle_n = \frac{1}{n}\sum_{\ell=1}^{n}\boldsymbol{r}(\boldsymbol{c}^\ell,\boldsymbol{g}^\ell), \tag{5.24}$$

where the masks $\{\boldsymbol{c}^\ell\}_{\ell=1}^n$ are selected as before, while $\{\boldsymbol{g}^\ell\}_{\ell=1}^n$ are the solutions to the corresponding tonal optimisation problems (5.16). Next we show that after relaxing assumptions on the grey values compared to Theorem 1, the MMSE estimate actually corresponds to DbI with an approximating inpainting instead of an interpolating one.

Proposition 2 (DbI with Approximating Inpainting as an MMSE Estimate)**.** *The DbI result based on a best approximating inpainting* (5.24) *can also be interpreted as an MMSE estimate, assuming that the grey values* $\boldsymbol{h}$ *are now also a random variable conditioned on* $\boldsymbol{f}$*.*

Proof. Firstly, we note that the minimisation problem for the MMSE now differs, as the expectation has to be taken over the grey values $\boldsymbol{h}$ as well:

$$\begin{aligned}\min_{\boldsymbol{u}\in\mathbb{R}^N} \mathbb{E}[\|\boldsymbol{u}-\boldsymbol{w}\|_2^2|\boldsymbol{f}] &= \min_{\boldsymbol{u}\in\mathbb{R}^N} \mathbb{E}[\|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})\|_2^2|\boldsymbol{f}] \\ &= \min_{\boldsymbol{u}\in\mathbb{R}^N} \sum_{\boldsymbol{c}\in\{0,1\}^N} \mathbb{E}[\|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})\|_2^2|\boldsymbol{f},\boldsymbol{c}]\, p(\boldsymbol{c}|\boldsymbol{f}) \\ &= \min_{\boldsymbol{u}\in\mathbb{R}^N} \sum_{\boldsymbol{c}\in\{0,1\}^N} \left(\int_{\boldsymbol{h}\in\mathbb{R}^N} \|\boldsymbol{u}-\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})\|_2^2\, p(\boldsymbol{h}|\boldsymbol{f},\boldsymbol{c})\, d\boldsymbol{h}\right) p(\boldsymbol{c}|\boldsymbol{f}).\end{aligned} \tag{5.25}$$

As before, differentiation w.r.t. $\boldsymbol{u}$ yields the MMSE estimate

$$\boldsymbol{u}^{\text{MMSE}} = \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f}] = \sum_{\boldsymbol{c}\in\{0,1\}^N} \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}]\, p(\boldsymbol{c}|\boldsymbol{f}), \tag{5.26}$$

which is similar to (5.23), but now contains the expectation

$$\mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}] = \int_{\boldsymbol{h}\in\mathbb{R}^N} \boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})\, p(\boldsymbol{h}|\boldsymbol{f},\boldsymbol{c})\, d\boldsymbol{h}. \tag{5.27}$$

To compute $\mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}]$, we need to know the a posteriori density $p(\boldsymbol{h}|\boldsymbol{f},\boldsymbol{c})$. If we assume that the noise is normally distributed $\boldsymbol{n} = (\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h}) - \boldsymbol{f}) \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\boldsymbol{n}}^2 \boldsymbol{I})$, and that the grey values restricted to the mask $\boldsymbol{h}|_{\boldsymbol{c}}$ are normally distributed $\boldsymbol{h}|_{\boldsymbol{c}} \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2 \boldsymbol{I})$, then the expectation can be calculated [145] as

$$\mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}] = \boldsymbol{B_c}\,\mathbb{E}[\boldsymbol{h}|_{\boldsymbol{c}}|\boldsymbol{f},\boldsymbol{c}] = \boldsymbol{B_c}\left(\frac{\sigma_{\boldsymbol{n}}^2}{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2}\boldsymbol{I} + \boldsymbol{B_c}^\top \boldsymbol{B_c}\right)^{-1} \boldsymbol{B_c}^\top \boldsymbol{f}. \tag{5.28}$$

Since we do not know $\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}$ and because the assumption of the normality of the grey values may not be a very plausible one, we can dispense away with it by taking $\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}} \to \infty$, which results in a tonally optimised inpainting:

$$\lim_{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}\to\infty} \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}] = \boldsymbol{B_c} \lim_{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}\to\infty} \left(\frac{\sigma_{\boldsymbol{n}}^2}{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2}\boldsymbol{I} + \boldsymbol{B_c}^\top \boldsymbol{B_c}\right)^{-1} \boldsymbol{B_c}^\top \boldsymbol{f} = \boldsymbol{B_c}(\boldsymbol{B_c})^+ \boldsymbol{f}. \tag{5.29}$$

Using $\boldsymbol{B_c}(\boldsymbol{B_c})^+\boldsymbol{f} = \boldsymbol{r}(\boldsymbol{c}, (\boldsymbol{B_c})^+\boldsymbol{f})$, the new MMSE estimate differs with (5.23) only in that we have approximation instead of interpolation:

$$\boldsymbol{u}^{\text{MMSE}} = \sum_{\boldsymbol{c}\in\{0,1\}^N} \mathbb{E}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f},\boldsymbol{c}]\, p(\boldsymbol{c}|\boldsymbol{f}) = \sum_{\boldsymbol{c}\in\{0,1\}^N} \boldsymbol{r}(\boldsymbol{c}, (\boldsymbol{B_c})^+\boldsymbol{f})\, p(\boldsymbol{c}|\boldsymbol{f}). \tag{5.30}$$

This corresponds exactly to the expectation of the approximating DbI formulation. □

We note that the above analysis did not require $\boldsymbol{r}$ to be linear in $\boldsymbol{f}$ except for the approximation of $\boldsymbol{f}$. Given a fixed $\boldsymbol{c}$, a natural extension to nonlinear operators could use nonlinear least-squares to compute something similar to $\boldsymbol{B}_{\boldsymbol{c}}^{+}\boldsymbol{f}$. By using the approximating formulation, we project the image onto the various sub-spaces induced by the inpainting operator $\boldsymbol{r}$ and the mask $\boldsymbol{c}$. We will show in Section 5.6.3.2 that in practice, tonal optimisation is able to improve quality and to reduce the variance of MMSE denoising, since it mitigates the error from the interpolation of noisy mask pixels and provides representations that are closer to $\boldsymbol{f}$ in terms of MSE.

5.3.1.2 Interpreting Tonal Optimisation as MAP Estimate

Not directly related to the classical averaging formulation of DbI, but nevertheless interesting and a valuable extension, is the fact that spatial and tonal optimisation for a single inpainting can also be framed as a maximum a posteriori (MAP) estimate. In MAP estimation, instead of minimising the MSE, we want to find an inpainting $\boldsymbol{w}$ that maximises the posterior:

$$\arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{f}) = \arg\max_{\boldsymbol{c},\boldsymbol{h}} p(\boldsymbol{h},\boldsymbol{c}|\boldsymbol{f}) = \arg\max_{\boldsymbol{c},\boldsymbol{h}} p(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{c})p(\boldsymbol{h}|\boldsymbol{c})p(\boldsymbol{c}). \tag{5.31}$$

We have assumed that $\boldsymbol{w} = \boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})$ is an injection, so we have $p(\boldsymbol{w}|\boldsymbol{f}) = p(\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})|\boldsymbol{f}) = p(\boldsymbol{h},\boldsymbol{c}|\boldsymbol{f})$. In the non-injective case one gets a set

$$p(\boldsymbol{w}|\boldsymbol{f}) = p(\boldsymbol{r}^{-1}(\boldsymbol{w})|\boldsymbol{f}) = p(\{\boldsymbol{h},\boldsymbol{c} : \boldsymbol{w} = \boldsymbol{r}(\boldsymbol{c},\boldsymbol{h})\}|\boldsymbol{f}), \tag{5.32}$$

which does not change the derivation meaningfully, except for introducing additional technical details. Thus, for the sake of clarity, we proceed with the injective case, but a similar argument holds in the general setting. The maximisation problem (5.31) can be split into two optimisation problems:

$$\max_{\boldsymbol{c},\boldsymbol{h}} p(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{c})p(\boldsymbol{h}|\boldsymbol{c})p(\boldsymbol{c}) = \max_{\boldsymbol{c}} \left(\max_{\boldsymbol{h}} p(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{c})p(\boldsymbol{h}|\boldsymbol{c}) \right) p(\boldsymbol{c}). \tag{5.33}$$

The inner one optimises over the grey values $\boldsymbol{h}$ given a mask $\boldsymbol{c}$, and the outer one optimises over the masks $\boldsymbol{c}$. If we again assume that $\boldsymbol{f} = \boldsymbol{r}(\boldsymbol{c},\boldsymbol{h}) + \boldsymbol{n}$, where $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\boldsymbol{n}}^2 \boldsymbol{I})$, then the density $p(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{c})$ is given by a Gaussian

$$p(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{c}) = \frac{1}{(2\pi\sigma_{\boldsymbol{n}}^2)^{N/2}} \exp\left(-\frac{\|\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h}) - \boldsymbol{f}\|_2^2}{\sigma_{\boldsymbol{n}}^2} \right). \tag{5.34}$$

Assuming also that the grey values are normally distributed, i.e., $\boldsymbol{h}|_{\boldsymbol{c}} \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2 \boldsymbol{I})$, then the minimisation problem w.r.t. $\boldsymbol{h}$ is what we call *the regularised tonal optimisation problem*:

$$\arg\max_{\boldsymbol{h}|_{\bar{\boldsymbol{c}}}=\boldsymbol{0}} \exp\left(-\frac{\|\boldsymbol{r}(\boldsymbol{c},\boldsymbol{h}) - \boldsymbol{f}\|_2^2}{\sigma_{\boldsymbol{n}}^2} - \frac{\|\boldsymbol{h}|_{\boldsymbol{c}}\|_2^2}{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2} \right) = \arg\min_{\boldsymbol{h}|_{\bar{\boldsymbol{c}}}=\boldsymbol{0}} \|\boldsymbol{B}_{\boldsymbol{c}}\boldsymbol{h}|_{\boldsymbol{c}} - \boldsymbol{f}\|_2^2 + \frac{\sigma_{\boldsymbol{n}}^2}{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2} \|\boldsymbol{h}_{\boldsymbol{c}}\|_2^2, \tag{5.35}$$

where the solution is the same as in (5.28):

$$\boldsymbol{h}|_{\boldsymbol{c}}^{*} = \left(\frac{\sigma_{\boldsymbol{n}}^2}{\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}}^2} \boldsymbol{I} + \boldsymbol{B}_{\boldsymbol{c}}^{\top} \boldsymbol{B}_{\boldsymbol{c}} \right)^{-1} \boldsymbol{B}_{\boldsymbol{c}}^{\top} \boldsymbol{f}. \tag{5.36}$$

Note that this can already be used for denoising with just a single inpainting with a mask $\boldsymbol{c}$, provided that we know the ratio of the variances of the noise and the grey values. The above expression suggests that we can then just apply a regularised tonal optimisation to get the best MAP estimate. As before, we may take $\sigma_{\boldsymbol{h}|_{\boldsymbol{c}}} \to \infty$ to get classical tonal optimisation if desired. Of course, we also need to optimise w.r.t. the masks according to $p(\boldsymbol{c})$. In fact, if we take $p(\boldsymbol{c}) = 0$ for $\|\boldsymbol{c}\|_0 \neq m$, and $p(\boldsymbol{c})$ being equal for all $\|\boldsymbol{c}\|_0 = m$, then we get the spatial optimisation problem with tonally optimised values:

$$\min_{\|\boldsymbol{c}\|_0 = m} \|\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{h}|_c^*(\boldsymbol{f})) - \boldsymbol{f}\|_2^2. \tag{5.37}$$

If we take the interpolating case, we get the classical spatial optimisation problem [156]:

$$\min_{\|\boldsymbol{c}\|_0 = m} \|\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f}) - \boldsymbol{f}\|_2^2. \tag{5.38}$$

The above further motivates using spatial optimisation for denoising in both the interpolation and approximation cases; see Section 5.5.

5.3.1.3 Bayesian Interpretation

In this subsection, we discuss how the above approaches fit in a general Bayesian perspective, which allows for meaningful interpretations of the occurring probabilities. This is valuable as MMSE and MAP estimates rely on a posterior $p(\boldsymbol{w}|\boldsymbol{f})$. Using Bayes' rule, this posterior can be rewritten as

$$p(\boldsymbol{w}|\boldsymbol{f}) = \frac{p(\boldsymbol{f}|\boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{f})} = \frac{p(\boldsymbol{f}|\boldsymbol{w})p(\boldsymbol{w})}{\int_{\mathbb{R}^N} p(\boldsymbol{f}|\boldsymbol{w})p(\boldsymbol{w})\, d\boldsymbol{w}}, \tag{5.39}$$

where $p(\boldsymbol{w})$ is the probability density function (PDF) for the distribution of images $\boldsymbol{w}$ from which we assume $\boldsymbol{f}_r$ to originate. The likelihood $p(\boldsymbol{f}|\boldsymbol{w})$ is the noise PDF, which in our case is a Gaussian. The term $p(\boldsymbol{f})$ is just a normalisation constant that is irrelevant in practice, since it is not a function of $\boldsymbol{w}$. This shows that the task of finding a proper posterior distribution corresponds to introducing an appropriate prior $p(\boldsymbol{w})$ under a given noise distribution $p(\boldsymbol{f}|\boldsymbol{w})$. This is known to be crucial for good denoising performance of Bayesian methods, and links our DbI framework to such approaches.

Incorporating the Inpainting Operator. To introduce an inpainting operator $\boldsymbol{r}$ into the above model, we make the assumption that any $\boldsymbol{w}$ is synthesised as $\boldsymbol{w} = \boldsymbol{r}(\boldsymbol{c}, \boldsymbol{h})$ for some mask $\boldsymbol{c}$ and some grey values $\boldsymbol{h}|_{\boldsymbol{c}}$. Since now the model depends on the masks we can rewrite the PDF as

$$p(\boldsymbol{w}|\boldsymbol{f}) = \sum_{\boldsymbol{c}\in\{0,1\}^N} p(\boldsymbol{w}|\boldsymbol{f},\boldsymbol{c})p(\boldsymbol{c}|\boldsymbol{f}), \tag{5.40}$$

which is where the conditional mask PMF $p(\boldsymbol{c}|\boldsymbol{f})$ comes into play – this is the other key ingredient for DbI along with the inpainting operator. We will see that this PMF allows us to introduce spatial adaptivity (Section 5.5.2, Figure 5.9) for operators that are otherwise not spatially adaptive. Finally, we can also rewrite $p(\boldsymbol{w}|\boldsymbol{f},\boldsymbol{c})$ using Bayes' rule in order to relate the above formulation to (5.39):

$$p(\boldsymbol{w}|\boldsymbol{f},\boldsymbol{c}) = \frac{p(\boldsymbol{f}|\boldsymbol{w},\boldsymbol{c})p(\boldsymbol{w}|\boldsymbol{c})}{p(\boldsymbol{f}|\boldsymbol{c})} = \frac{p(\boldsymbol{f}|\boldsymbol{w},\boldsymbol{c})p(\boldsymbol{w}|\boldsymbol{c})}{\int_{\mathbb{R}^N} p(\boldsymbol{f}|\boldsymbol{w},\boldsymbol{c})p(\boldsymbol{w}|\boldsymbol{c})\,d\boldsymbol{w}}. \tag{5.41}$$

This provides a similar interpretation, but now we have knowledge about the mask. As before $p(\boldsymbol{f}|\boldsymbol{w},\boldsymbol{c})$ models the noise, but now $p(\boldsymbol{w}|\boldsymbol{c})$ models the distribution of the grey values defining $\boldsymbol{w}$ given $\boldsymbol{c}$, i.e., the distribution of $\boldsymbol{h}|_{\boldsymbol{c}}$. As before, the denominator is a normalisation constant that is not practically relevant.

The Mask Posterior. Bayes' rule allows us to explore further theoretical considerations about the involved mask probabilities. We can study the mask posterior $p(\boldsymbol{c}|\boldsymbol{f})$ in more detail, using

$$p(\boldsymbol{c}|\boldsymbol{f}) = \frac{p(\boldsymbol{f}|\boldsymbol{c})p(\boldsymbol{c})}{p(\boldsymbol{f})}. \tag{5.42}$$

Now $p(\boldsymbol{c})$ models the probability of the mask $\boldsymbol{c}$ being generated (irrespective of $\boldsymbol{f}$) and $p(\boldsymbol{f}|\boldsymbol{c})$ models some measure of the noise and image content in relation to the mask. In practice, ideally the density $\mathbf{1}^\mathsf{T}\mathbb{E}[\boldsymbol{c}]/N$ should be chosen to be inversely proportional to the standard deviation of the noise. Similarly if we know that the noise distribution is space-variant, or if we suspect that features (e.g. edges) are present, we can choose the local density of $\boldsymbol{c}$ to account for that: higher for more prominent edges, lower for higher noise variance. The weight of these choices is modelled by $p(\boldsymbol{f}|\boldsymbol{c})$. Selecting $p(\boldsymbol{c})$ is less trivial, as it needs to match the mask distribution of natural images, i.e., the distribution of natural images from the perspective of the masks used in the inpainting operator. It is simpler to choose it based on the density, i.e., $p(\boldsymbol{c}) = p(\|\boldsymbol{c}\|_0/N)$, which makes it blind to spatial variations, or to just choose it as a constant, if we have no data on it. Note that these considerations are meant to provide a different view on the mask posterior and an alternative strategy on how to construct it. The adaptive mask selection methods that we consider in this work directly induce a mask posterior $p(\boldsymbol{c}|\boldsymbol{f})$ and do not model $p(\boldsymbol{f}|\boldsymbol{c})$ or $p(\boldsymbol{c})$. They are based on strategies from the noise-free case in image inpainting and we adapt and extend them to the noisy case. For all the approaches that we consider, we state their induced PMFs $p(\boldsymbol{c}|\boldsymbol{f})$ (see Equation (5.64), Theorem 5, Appendix A.2).

On the Importance of the Inpainting Operator. A crucial question is whether an inpainting operator $\boldsymbol{r}$ is suitable for modelling natural images in a sparse and robust manner, such that noise can be attenuated by averaging multiple nearby representations of a noisy image from a lower-dimensional image manifold. For $\boldsymbol{r}$ being homogeneous diffusion inpainting, we know that it has been used successfully for image compression of natural images with low to medium frequencies [155]. Moreover, we present new results in Section 5.4 that relate the MMSE estimate to homogeneous diffusion denoising. The large body of literature on sparse image approximation and compression should provide a reasonable selection of good inpainting operators $\boldsymbol{r}$. In the current work we also consider biharmonic inpainting (see Section 5.6.4).

Interplay between the Mask PMF and Homogeneous Diffusion. The basis vectors $\boldsymbol{B_c}$ for homogeneous diffusion are generally low-frequent and smooth, with the local frequency depending on the local density of the mask points. For a constant PMF $p(\boldsymbol{c}|\boldsymbol{f})$, i.e., a homogeneous mask density, we get a process similar to isotropic homogeneous diffusion, and it is in fact approximately equivalent to it, as we demonstrate later in Section 5.4.2 and Section 5.4.3. As such it also shares its drawbacks, i.e., smoothing equally over image structures and noise. More sophisticated denoising methods such as space-variant diffusion allow for steering the smoothing away from image structures by relying on a guidance image, e.g., the gradient magnitude $|\nabla u|$. Similarly, we may use the PMF $p(\boldsymbol{c}|\boldsymbol{f})$ to guide the denoising. One instance of a PMF that we consider is inspired by a result for mask selection in inpainting. Belhachmi et al. [28] have argued that the local density of an optimal inpainting mask $\boldsymbol{c}$ should be proportional to the pixelwise magnitude of the Laplacian $|\boldsymbol{Lf}|$. In our setting this translates to constructing a PMF p such that $\mathbb{E}[\boldsymbol{c}|\boldsymbol{f}] \sim |\boldsymbol{Lf}|$; see Section 5.5.2.

5.3.1.4 Convergence

A question which arises is how well the estimator $\langle \boldsymbol{u}\rangle_n$ approximates the MMSE estimate $\boldsymbol{u}^{\text{MMSE}} = \mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]$ as a function of the number of samples n. We consider this scaling behaviour in the next proposition.

Proposition 3 (Convergence of the DbI Estimator)**.** *The root mean square error (RMSE) $\sqrt{\text{MSE}(\langle \boldsymbol{u}\rangle_n, \mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}])}$ between the estimator $\langle \boldsymbol{u}\rangle_n$ and its expectation $\mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]$ scales as $O(n^{-1/2})$, where n is the number of sampled masks.*

Proof. We first recall that we can decompose the MSE between some estimator $\hat{\boldsymbol{\theta}}$ and some fixed parameter $\boldsymbol{\theta}$ into a variance and a bias part:

$$\begin{aligned}\text{MSE}(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}) &= \mathbb{E}\left[\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2^2\right] \\ &= \mathbb{E}\left[\|\hat{\boldsymbol{\theta}} - \mathbb{E}[\hat{\boldsymbol{\theta}}]\|_2^2\right] + \|\mathbb{E}[\hat{\boldsymbol{\theta}}] - \boldsymbol{\theta}\|_2^2 \\ &= \mathbb{V}[\hat{\boldsymbol{\theta}}] + \text{Bias}(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta})^2.\end{aligned} \tag{5.43}$$

If we consider the MSE between the estimator $\langle \boldsymbol{u}\rangle_n$ and its expectation $\mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]$, the bias vanishes and we have $\text{MSE}(\langle \boldsymbol{u}\rangle_n, \mathbb{E}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]) = \mathbb{V}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]$. The variance $\mathbb{V}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}]$ is given by

$$\mathbb{V}[\langle \boldsymbol{u}\rangle_n|\boldsymbol{f}] = \mathbb{V}\left[\frac{1}{n}\sum_{\ell=1}^{n} \boldsymbol{r}(\boldsymbol{c}^\ell, \boldsymbol{f})\middle|\boldsymbol{f}\right] = \frac{1}{n^2}\sum_{\ell=1}^{n} \mathbb{V}\left[\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})|\boldsymbol{f}\right] = \frac{1}{n}\mathbb{V}[\boldsymbol{r}(\boldsymbol{c}, \boldsymbol{f})|\boldsymbol{f}]. \tag{5.44}$$

The second equality holds because the masks are independent and identically distributed. For a finite variance $\mathbb{V}[\boldsymbol{r}(\boldsymbol{c},\boldsymbol{f})|\boldsymbol{f}]$, the root mean square error between the estimator and its expectation thus scales as $O(n^{-1/2})$. □

5.3.1.5 Acceleration by Low-Discrepancy Sequences

When the masks are random variables, as noted in Section 5.3.1.4, we have a somewhat slow convergence of $O(n^{-1/2})$. Informally this means that to decrease the RMSE by a factor 4 we would need 16 times as many samples. The natural question arises whether we can do better by trading randomness for a more structured sampling strategy. The answer is positive, as in the context of integration (and our problem can be framed as such w.r.t. the counting measure), a prominent approach for speeding up convergence is the use of low-discrepancy sequences. These sequences fill up space more uniformly than random sequences. The uniformity is typically quantified using the (star) discrepancy of the sequence. Theoretically, the Koksma-Hlawka inequality [140] allows one to bound the numerical integration error, i.e., $\|\langle\boldsymbol{u}\rangle_n - \mathbb{E}[\langle\boldsymbol{u}\rangle_n|\boldsymbol{f}]\|_2$ in our case, by using the product of the discrepancy of the sequence and the variation of the integrand. In practice this usually translates to a convergence that can reach as high as $O(n^{-1})$ which is much better than the $O(n^{-1/2})$ convergence for the purely random case. Experimental results illustrating a boost to the convergence in the DbI setting are presented in Section 5.6.2.

5.4 Linking Denoising by Inpainting to Homogeneous Diffusion

The simplest approaches for mask selection in the DbI framework are those that are independent of the image that is to be filtered ($p(\boldsymbol{c}|\boldsymbol{f}) \equiv p(\boldsymbol{c})$). We consider shifted regular masks as well as randomly selected masks. They are characterised by a spatially flat expectation $\mathbb{E}[\boldsymbol{c}] = \text{const}$. In the following, we briefly introduce regular masks, show how they can be used in the DbI framework and discuss the resulting filtering behaviour. Then we derive relations between DbI with regular masks and homogeneous diffusion filtering in 1D. Afterwards, using random masks instead of regular masks, we empirically extend those results to the 2-D setting.

5.4.1 Regular Masks

Regular masks are created by generating a pattern with each r-th pixel in x- and each s-th pixel in y-direction being added to the mask. We can then shift such a mask in both directions to obtain multiple masks. If we assume an $n_x \times n_y$ pixel grid, we can create such a regular mask via

$$c_{i,j} = \begin{cases} 1 & \text{if } i \bmod r = 0 \text{ and } j \bmod s = 0, \\ 0 & \text{else.} \end{cases} \tag{5.45}$$

We have s options of shifting this regular mask in x-direction and r options in y-direction, adding up to $n = rs$ total possible configurations. Denoting by $p \in \{0, \dots, r-1\}$ and $q \in \{0, \dots, s-1\}$ the shift in x- and y-direction, respectively, we can write the shifted masks as

$$c_{i,j}^{ps+q+1} = \begin{cases} 1 & \text{if } i \bmod r = p \text{ and } j \bmod s = q, \\ 0 & \text{else.} \end{cases} \tag{5.46}$$

Clearly, the created masks are independent of the image. Furthermore, the mask density is constant over the entire image, leading to the same smoothing strength at all locations, solely determined by the total mask density, i.e., by the spacing. If $r = s$, this smoothing is equally strong in x- and y-direction. Visually one then observes a smoothing behaviour that resembles the one of homogeneous diffusion filtering (see Figure 5.2(c) and Figure 5.2(d)). The influence of the mask density on the smoothing strength can be observed in Figure 5.2(d) and Figure 5.2(e).

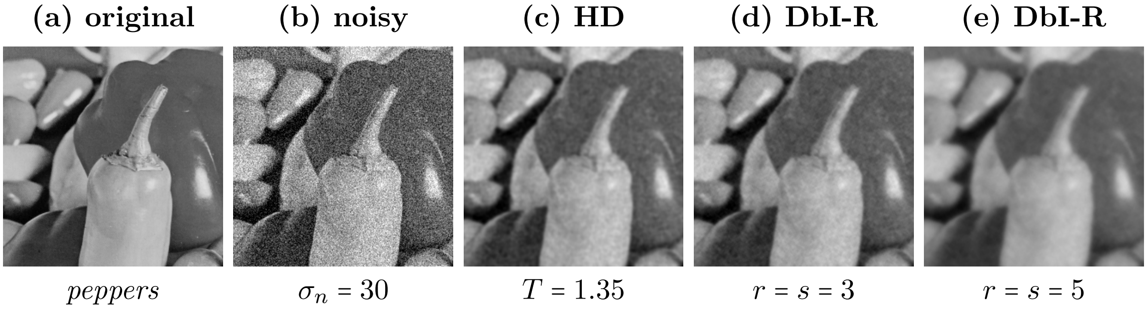


Figure 5.2: Comparison of homogeneous diffusion (HD) and denoising by inpainting with regular masks (DbI-R) on the test image *peppers* with $\sigma_n = 30$. Figure 5.2(c) and Figure 5.2(d) show the visual similarities of both methods. Figure 5.2(d) and Figure 5.2(e) illustrate the influence of the expected density $\mathbf{1}^\top \mathbb{E}[\boldsymbol{c}]/N$ on the smoothness of the reconstruction: Figure 5.2(e) was intentionally chosen with a density that is too low resulting in too much smoothing.

The similarity between the methods can not only be observed visually, but also established theoretically: Next we provide a derivation in the 1-D case for regular masks relating the diffusion time of homogeneous diffusion to the mask density in DbI.

5.4.2 Mathematical Analysis in 1D

We consider a discrete 1-D signal $\boldsymbol{f}$ and regular inpainting masks with spacing r and shift $p \in \{0, \dots, r-1\}$. It is known that in 1D, homogeneous diffusion inpainting and linear interpolation are equivalent. Thus, an inpainted pixel at position i can be described in terms of its two neighbouring mask pixels. We denote the distance between the pixel i and its neighbouring mask pixel on the left by $\ell \coloneqq |i-p| \bmod r$, which implies that for mask pixels we have $\ell = 0$. Accordingly, the distance to the mask pixel on the right is given by $r - \ell$. The interpolated value at pixel i for mask $\ell + 1$ is then

$$v_i^{\ell+1} = \frac{r-\ell}{r} f_{i-\ell} + \frac{\ell}{r} f_{i+r-\ell}. \tag{5.47}$$

To obtain the final result, the inpaintings from the r shifted masks are averaged. We get

$$\begin{aligned} u_i = \frac{1}{r}\sum_{\ell=0}^{r-1} v_i^{\ell+1} &= \frac{1}{r}\left(f_i + \sum_{\ell=1}^{r-1}\frac{r-\ell}{r} f_{i-\ell} + \frac{\ell}{r} f_{i+r-\ell}\right) \\ &= \frac{1}{r^2}\left(r f_i + \sum_{\ell=1}^{r-1}\ell\left(f_{i-(r-\ell)} + f_{i+(r-\ell)}\right)\right), \end{aligned} \tag{5.48}$$

where the last line reveals the general form of the filter in dependence of the spacing r: The filter is given by a hat kernel with central weight $1/r$ and width $2r-1$. In Theorem 4 we demonstrate that this kernel can be seen as a consistent discretisation of $\partial_t u = \partial_{xx} u$. Consequently, convolution with such a kernel approximates Gaussian smoothing, which explains the visual similarity of the results in Figure 5.2. Since the spacing r determines the size of the smoothing kernel, we explicitly see the connection between the mask density and the smoothing strength. For the special case of $r = 2$, (5.48) yields

$$u_i = \frac{f_{i-1} + 2f_i + f_{i+1}}{4}, \tag{5.49}$$

which is exactly a single step of an explicit scheme for homogeneous diffusion with step size $T = \frac{1}{4}$ and initial signal $\boldsymbol{f}$ (assuming grid size $h = 1$). If we reformulate (5.48) in a way that resembles an explicit scheme for homogeneous diffusion, we can derive a general connection between the spacing r (and thus the density) of denoising by inpainting with regular masks and the time step size of such an explicit scheme, which we state in Theorem 4.

Theorem 4 (Connection between Mask Density and Diffusion Time)**.** *Given the r shifted regular inpainting masks in 1D, each of density $d = 1/r$, denoising by inpainting approximates explicit homogeneous diffusion at time*

$$\boxed{T = \frac{1-d^2}{12d^2}} \tag{5.50}$$

Proof. In (5.48) we derived the general form of the filter corresponding to denoising by inpainting with regular masks of spacing r as

$$u_i = \frac{1}{r^2}\left(r f_i + \sum_{\ell=1}^{r-1}\ell\left(f_{i-(r-\ell)} + f_{i+(r-\ell)}\right)\right). \tag{5.51}$$

We can rewrite this in the following manner:

$$\begin{aligned} u_i &= \frac{1}{r^2}\left(r f_i + \sum_{\ell=1}^{r-1}\ell\left(f_{i-(r-\ell)} + f_{i+(r-\ell)}\right)\right) \\ &= \frac{1}{r^2}\left(r^2 f_i - 2\sum_{\ell=1}^{r-1}\ell f_i + \sum_{\ell=1}^{r-1}\ell\left(f_{i-(r-\ell)} + f_{i+(r-\ell)}\right)\right) \\ &= f_i + \frac{1}{r^2}\sum_{\ell=1}^{r-1}\ell\left(f_{i-(r-\ell)} - 2f_i + f_{i+(r-\ell)}\right), \end{aligned} \tag{5.52}$$

where we have used that $\sum_{\ell=1}^{r-1} \ell = \frac{(r-1)r}{2}$. Then we may write

$$\begin{aligned} u_i - f_i &= \frac{1}{r^2} \sum_{\ell=1}^{r-1} \ell \left(f_{i-(r-\ell)} - 2f_i + f_{i+(r-\ell)} \right) \\ &= \frac{1}{r^2} \sum_{\ell=1}^{r-1} \ell (r-\ell)^2 \, \frac{f_{i-(r-\ell)} - 2f_i + f_{i+(r-\ell)}}{(r-\ell)^2} \\ &= \sum_{\ell=1}^{r-1} \frac{\ell (r-\ell)^2}{r^2} \, \frac{f_{i-(r-\ell)} - 2f_i + f_{i+(r-\ell)}}{(r-\ell)^2} . \end{aligned} \tag{5.53}$$

By approximating $f_{i\pm(r-\ell)}$ via a Taylor expansion and using the sampling distance h, we can derive the time step size as

$$\begin{aligned} u_i - f_i &= \sum_{\ell=1}^{r-1} \frac{\ell (r-\ell)^2}{r^2} \, \frac{f_{i-(r-\ell)} - 2f_i + f_{i+(r-\ell)}}{(r-\ell)^2} \\ &= \sum_{\ell=1}^{r-1} \left(\frac{\ell (r-\ell)^2}{r^2} \right) \left(h^2 \, d_{xx} f\big|_i + \frac{h^4 (r-\ell)^2}{12} \, d_{xxxx} f\big|_i + \mathcal{O}(h^6) \right) \\ &= h^2 \sum_{\ell=1}^{r-1} \left(\frac{\ell (r-\ell)^2}{r^2} \right) \left(d_{xx} f\big|_i + \mathcal{O}(h^2) \right) \\ &\approx h^2 \sum_{\ell=1}^{r-1} \left(\frac{\ell (r-\ell)^2}{r^2} \right) d_{xx} f\big|_i . \end{aligned} \tag{5.54}$$

We end up with an approximation of an explicit scheme with time step size

$$T = h^2 \sum_{\ell=1}^{r-1} \frac{\ell (r-\ell)^2}{r^2} = \frac{h^2 (r^2 - 1)}{12} . \tag{5.55}$$

Using that the density is the inverse of the grid spacing and setting $h = 1$, we derive the final relation between T and the density d, given by

$$T = \frac{1 - d^2}{12 d^2} . \tag{5.56}$$

□

5.4.3 Empirical Extension to 2D

To derive the relationship to the diffusion time in the 1-D case we used the fact that the solution of the Laplace equation with Dirichlet boundaries is given by linear interpolation. That is, we know the closed form of the inpainting echoes in 1D. In 2D a closed form solution for those is not known, however they may be computed numerically. Thus our goal is to establish a relationship between the diffusion time and the density numerically.

We take as a starting point the ansatz from the 1-D case that the diffusion time T is given as $\frac{1-d^2}{12d^2}$, but generalise it to the form $T \approx \frac{1-d^\gamma}{\beta d^\gamma}$. Provided that this conjecture is correct we only need to find the constants β and γ. Since regular masks only allow for a stepwise adaptation of the mask density, they are not well-suited for generating a large number of data points at different densities. Therefore, we use uniform random masks instead, which also have a spatially flat expectation, i.e., $\mathbb{E}[\boldsymbol{c}] = \text{const}$.

First we numerically tabulate the relationship between the density and the diffusion time. That is, given a density d we find the diffusion time $T(d)$ which minimises the difference between the filter matrices:

$$T(d) = \underset{T \geq 0}{\arg\min} \, \|\boldsymbol{A}_{DbI}(d) - \boldsymbol{A}_{HD}(T)\|_F^2. \tag{5.57}$$

Here $\|\cdot\|_F$ is the Frobenius norm, and the matrices are the DbI filter matrix resulting from a probability mass function for masks with expected density d, and the matrix modelling homogeneous diffusion at time T using an implicit Euler discretisation:

$$\boldsymbol{A}_{DbI}(d) \coloneqq \mathbb{E}\left[(\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{C})\,\boldsymbol{L})^{-1}\,\boldsymbol{C}\right], \quad \frac{1}{N}\boldsymbol{1}^{\mathsf{T}}\mathbb{E}[\boldsymbol{c}] = d, \tag{5.58}$$

$$\boldsymbol{A}_{HD}(T) \coloneqq (\boldsymbol{I} + T\boldsymbol{L})^{-1}. \tag{5.59}$$

We estimate $\boldsymbol{A}_{DbI}$ using 1024 sampled masks. Then, having the relationship $d \mapsto T(d)$ we find that $T(d) \approx \frac{1-d^\gamma}{\beta d^\gamma}$ for $\beta = 4.58$, $\gamma = 1.3$, which is illustrated in Figure 5.3. Note the high quality of the data fit, which confirms the accuracy of the derived relation.

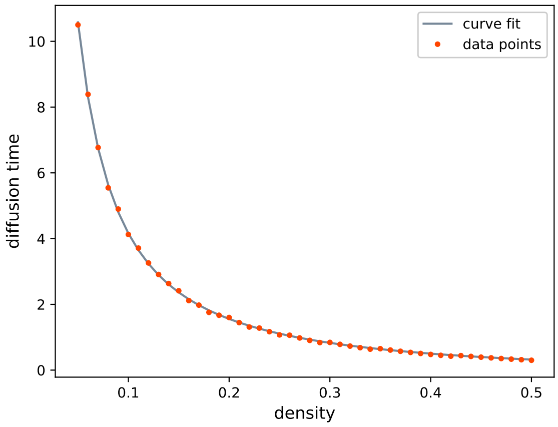


FIGURE 5.3: The fit based on the ansatz $\frac{1-d^\gamma}{\beta d^\gamma}$ with $\beta = 4.58$, $\gamma = 1.3$ and the tabulated correspondence between density and diffusion time. The results are obtained with denoising by inpainting with uniform random masks and an implicit scheme for homogeneous diffusion. They show that also in 2D our ansatz captures the relation between mask density and diffusion time very accurately.

5.5 Spatial Optimisation for Denoising by Inpainting

As we have seen in Section 5.4, the use of non-adaptive masks restricts the DbI framework, as it entails a non-adaptive smoothing behaviour. Furthermore, our results from Section 5.3.1 emphasise the importance of spatial optimisation in the context of image denoising. In [2], an adaptive mask selection approach enables the framework to perform edge-preserving image filtering, although the simple homogeneous diffusion inpainting operator by itself is space-invariant. This approach thus implies a different paradigm for image denoising: *Instead of optimising the denoising operator, one can optimise the data.* In this section, we will first present the strategy that was proposed in [2]. Then we propose an alternative, simpler approach that eventually gains its power by the application of tonal optimisation.

5.5.1 Densification Method

Two well-known mask selection strategies from image compression are probabilistic sparsification [156] and densification [112], which build the mask in an iterative way using a top-down and a bottom-up strategy, respectively.

In probabilistic sparsification, we start with a full mask and take away the least important pixels from a number of randomly selected candidates in each iteration. To identify those pixels, we temporarily exclude all candidates from the mask and compute an inpainting. Then the candidate locations with the highest local (i.e., pixelwise) reconstruction error are added back to the mask as they are assumed to be the most important, while the others remain permanently excluded. This process is repeated until the desired mask density is reached. In probabilistic densification, the initial mask is empty and again a number of candidate pixels are selected. Given an inpainting with the current mask (in the first step some pixels have to be chosen at random) we select and add those candidates to the mask that have the highest local reconstruction error.

In the noisy setting, special care is required as the pixel selection based on the *local* reconstruction error is not reliable. The local error does not allow the algorithm to distinguish between noise and important image structures, such as edges. If a pixel contains strong noise, this creates a large local error because – just like edges – the noise cannot be reconstructed by the smooth inpainting. Introducing such a noisy pixel into the mask is not desirable. We cure this problem by judging the importance of a pixel based on its effect on the *global* reconstruction error. We do this by calculating a full inpainting for each candidate pixel. While this improves the quality of the selected mask, it drastically increases the run time.

Even though in the noise-free setting densification and sparsification yield results of comparable quality [111], this is different when handling noisy data. For sparsification we initially have very dense masks. If we exclude candidate pixels from such masks, the reconstructions often only differ at the locations of these pixels. Therefore, sparsification tends to keep noisy pixels in the mask, even when a global reconstruction error is computed. This problem does not occur in probabilistic densification, as for a sparse mask, the candidate pixels have a global influence. The result of this effect is illustrated in Figure 5.4. Here, densification is able to select appropriate pixels that lead to an almost perfect result while sparsification fails to reconstruct the image properly.

Thus, we opt for a *probabilistic densification* algorithm based on a *global* error computation, which is described in Algorithm 3 and has been proposed in [2]. An additional advantage of this probabilistic densification method is that it does not only select pixels at useful locations (e.g., close to edges), but also implicitly avoids picking pixels that are too noisy, as they would have a negative impact on the reconstruction quality. The method can be interpreted according to the probabilistic mask generation framework from Section 5.3, and we provide the implied mask probabilities in the following proposition.

Proposition 5 (Mask Probabilities implied by the Densification Method)**.** *A mask* $\boldsymbol{c}$ *generated by probabilistic densification has the conditional probability density function*

$$p(\boldsymbol{c}|\boldsymbol{f}) = \sum_{\sigma \in S_m} p_\sigma(\boldsymbol{c}|\boldsymbol{f}), \tag{5.60}$$

where $m = \|\boldsymbol{c}\|_0$ *is the number of mask pixels, the sum is taken over the group* S_m *of permutations of the ordering of the* m *mask pixels, and* $p_\sigma(\boldsymbol{c}|\boldsymbol{f})$ *denotes the probability that the* m *mask points were introduced in the order* σ*. The latter is the product of the probabilities of selecting one mask pixel at each step:*

$$p_\sigma(\boldsymbol{c}|\boldsymbol{f}) = p_\sigma^m(\boldsymbol{c}|\boldsymbol{f}) \dots p_\sigma^1(\boldsymbol{c}|\boldsymbol{f}). \tag{5.61}$$

The probability of picking the k*-th mask pixel (according to the permutation* σ*) at step* k *has the following form:*

$$p_\sigma^k(\boldsymbol{c}|\boldsymbol{f}) = \sum_{\beta=1}^{\alpha} \frac{1}{\beta} \frac{\binom{N_{eq}-1}{\beta-1}\binom{N_{gt}}{\alpha-\beta}}{\binom{N-k}{\alpha}}, \tag{5.62}$$

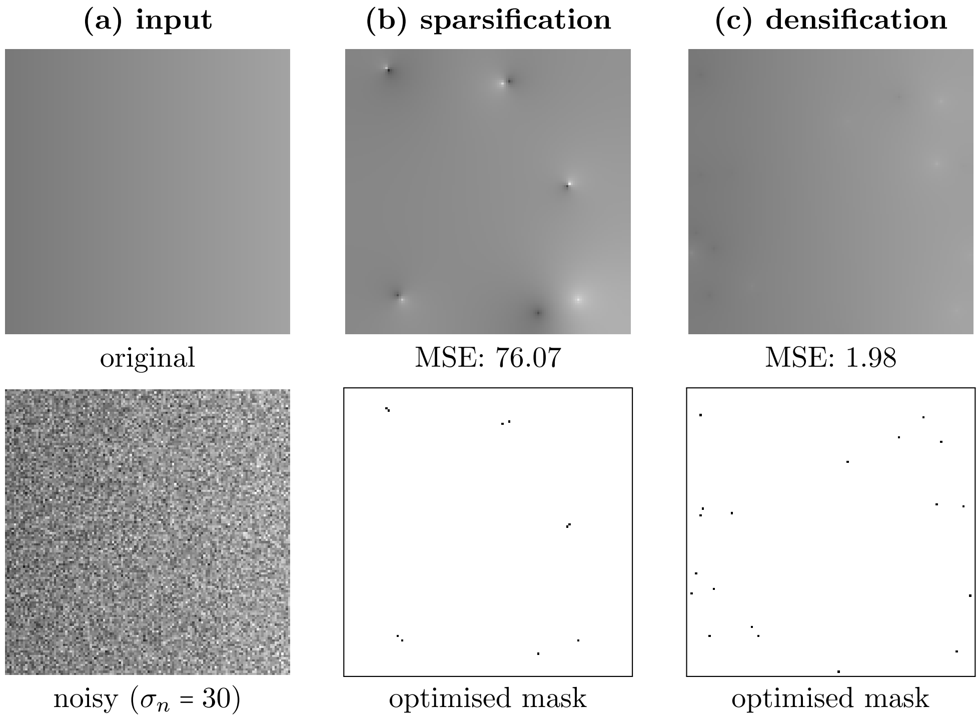
where α *is the number of candidates considered per step,* N_{gt} *is the number of non-mask pixels at step* k *that would have resulted in an inpainting with a higher MSE if they were chosen instead of the* k*-th mask pixel in* σ*, and* N_{eq} *is the number of non-mask pixels that would have resulted in the same MSE.*

Proof. We present the proof of this result in Appendix A.1. □

5.5.2 Acceleration via the Analytic Results of Belhachmi et al.

As the global error computation in the previous approach requires calculating an inpainting for each candidate pixel, the run time is substantial. Therefore, we propose another approach, with the goal of a faster mask generation process. We refer to this method as the *analytic method*. It is based on the results of Belhachmi et al. [28]. They have shown that the mask density for homogeneous diffusion inpainting should be proportional to the pointwise magnitude of the Laplacian $|\boldsymbol{L}\boldsymbol{f}|$. Additionally, they suggest using the Gaussian-smoothed version $\boldsymbol{f}_\sigma := \boldsymbol{K}_\sigma * \boldsymbol{f}$ of $\boldsymbol{f}$ even in the noise-free setting. Here $\boldsymbol{K}_\sigma$ is a discrete approximation of a Gaussian with standard deviation σ. This step proves even more beneficial in our setting, since we are calculating the Laplacian of noisy data, and regularising $\boldsymbol{f}$ helps considerably for constructing a reasonable guidance image $|\boldsymbol{L}\boldsymbol{f}_\sigma|$.

As we require multiple different binary masks for our framework, we sample from $|\boldsymbol{L}\boldsymbol{f}_\sigma|$ by using a simple and fast Poisson sampling. Given a density image $\boldsymbol{d} \in [0,1]^N$, we can sample a mask according to it by generating a uniform random number $v_i \sim U[0,1]$ for



(a) input **(b) sparsification** **(c) densification**

original MSE: 76.07 MSE: 1.98

noisy ($\sigma_n = 30$) optimised mask optimised mask

FIGURE 5.4: Comparison of sparsification and densification on a synthetic test image with $\sigma_n = 30$ [2]. For both methods, the mask density d was optimised with a grid search w.r.t. the MSE. The noisy gradient image is not reconstructed adequately by sparsification, since it favours keeping noisy pixels in the first iterations due to localisation. Densification does not suffer from this problem and thereby achieves a better denoised reconstruction.

Algorithm 3. Mask densification with global error computation [2].

```
Input: Noisy image $\boldsymbol{f} \in \mathbb{R}^N$, number of candidates $\alpha$, desired final mask density $d$.

Initialisation: Mask $\boldsymbol{c} = 0$ is empty.

Compute:
    do
        1. Choose randomly a set $A \subset \{k \in \{1, ..., N\} \,|\, c_k = 0\}$ with $\alpha$ candidates.
           for all $i \in A$ do
               2. Set temporary mask $\boldsymbol{m}^i$ such that $\forall k \in \{1, ..., \alpha\} \setminus \{i\} : m^i_k = c_k$, $m^i_i = 1$.
               3. Compute reconstruction $\boldsymbol{u}^i$ from mask $\boldsymbol{m}^i$ and image data $\boldsymbol{f}$.
           end for
        4. Set $\boldsymbol{c} = \underset{\boldsymbol{m}^i,\, i \in A}{\arg\min}\, \mathrm{MSE}(\boldsymbol{u}^i, \boldsymbol{f})$. This adds one mask point to $\boldsymbol{c}$.
    while pixel density of $\boldsymbol{c}$ smaller than $d$.

Output: Mask $\boldsymbol{c}$ of density $d$.
```

each pixel i and then thresholding at d_i:

$$c_i = \begin{cases} 1 & \text{if } v_i \leq d_i, \\ 0 & \text{if } v_i > d_i. \end{cases} \tag{5.63}$$

Then the probability mass function $p_{\boldsymbol{d}}$ for sampling a mask $\boldsymbol{c}$ given the density image $\boldsymbol{d}$ is

$$p_{\boldsymbol{d}}(\boldsymbol{c}) = \frac{1}{P}\prod_{i=1}^{N}(d_i)^{c_i}(1-d_i)^{1-c_i}, \quad P = \sum_{\boldsymbol{c}\in\{0,1\}^N}\prod_{i=1}^{N}(d_i)^{c_i}(1-d_i)^{1-c_i}. \tag{5.64}$$

By construction the mask would have an expected density equal to the mean value of $\boldsymbol{d}$. In our approach we set the per pixel probabilities to

$$\boldsymbol{d} = \min\left\{C|\boldsymbol{L}\boldsymbol{f}_\sigma|, \boldsymbol{1}\right\}, \tag{5.65}$$

where the minima are taken pointwise, and C is a constant chosen such that the mean value of $\boldsymbol{d}$ is equal to the desired mask density. Figure 5.5 shows the pipeline for mask generation with this method. One can observe in Figure 5.5(b) that $\boldsymbol{d}$ is strongly affected by the noise despite the pre-smoothing. This is because we calculate second-order derivatives that are even more sensitive to noise. When sampling from this image the mask is drawn towards noisy pixels. To counteract this, we propose to perform an additional outer smoothing of the probability image $\boldsymbol{d}$, after the absolute value of the Laplacian is taken, thus modifying it to

$$\boldsymbol{d} = \min\left\{C\left(\boldsymbol{K}_\rho * |\boldsymbol{L}\boldsymbol{f}_\sigma|\right), \boldsymbol{1}\right\}, \tag{5.66}$$

with a post-smoothing parameter ρ. Our proposed selection strategy offers an instant generation of adaptive masks, in a sense that it does not require the calculation of any inpainting. Furthermore, it provides a transparent formulation of the mask PMF (see (5.64)) and as such exhibits a specifically simple interpretation in the context of our probabilistic framework in Section 5.3.1. On the other hand, contrary to probabilistic densification it does not have a mechanism to avoid noisy mask pixels. To obtain the best possible results, the pre-smoothing parameter σ, the post-smoothing parameter ρ, and the desired mask density have to be optimised depending on the image content and the noise level.

Note that Belhachmi et al. [28] apply Floyd-Steinberg dithering [85], which includes an error diffusion in the binarisation process. This strategy can be equipped with a random component in order to generate multiple masks, which makes it an alternative to Poisson sampling for us. We have tested both methods and found that there is no advantage in using Floyd-Steinberg dithering. Thus, we opt for the simple Poisson sampling. Nonetheless, we give the mask probabilities for sampling with error diffusion methods in Appendix A.2.

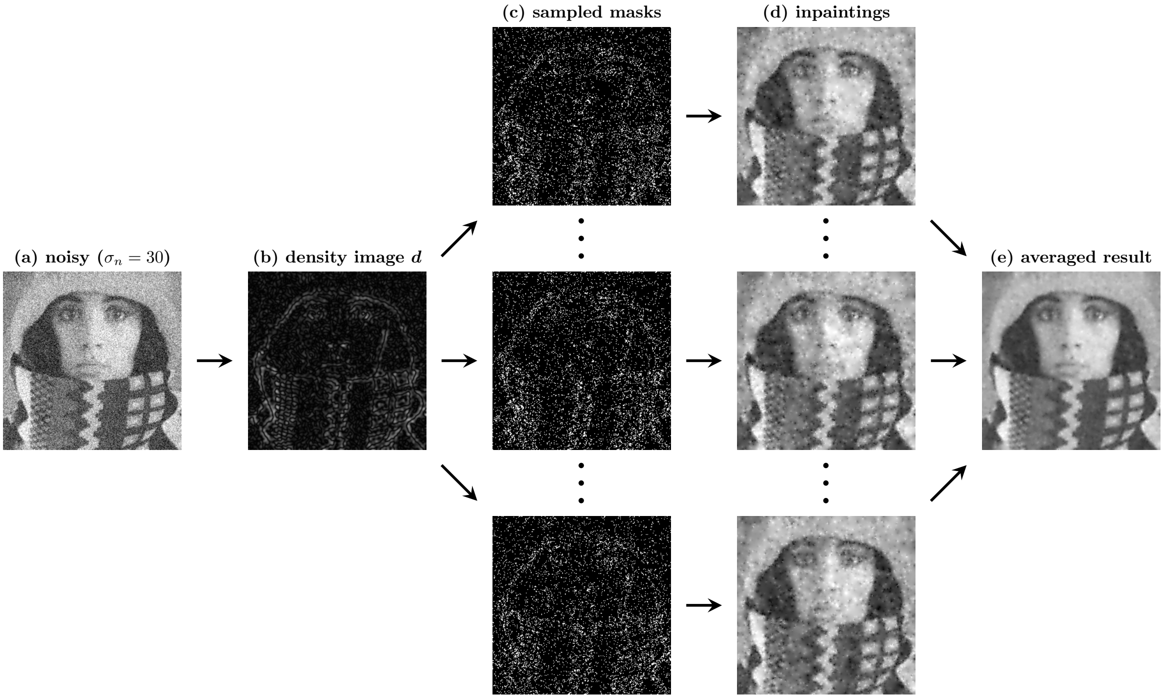


FIGURE 5.5: Pipeline for mask generation with the analytic method. (a) Test image *trui* with $\sigma_n = 30$. (b) Target image (without post-smoothing) from which masks are sampled. (c) Three examples of Poisson-sampled masks. (d) Corresponding homogeneous diffusion inpaintings. (e) Averaged inpaintings (from 32 masks), final denoising result.

5.6 Experiments

In this section, we present our experiments. They evaluate our theories and compare the different DbI strategies in practice. Firstly, we confirm the accuracy of the 1-D relation that we derived for DbI with regular mask in Section 5.4.2. We also display the corresponding results in 2D. Next, we show that the theoretical convergence estimates from Section 5.3.1.4 also hold in practice and evaluate the gain through low-discrepancy-based sampling (see Section 5.3.1.5). Furthermore, we assess the spatial and tonal mask optimisation approaches. To this end, we compare DbI to PDE-based methods of similar structural complexity. Aside from homogeneous diffusion, we choose linear space-variant diffusion and nonlinear diffusion as representatives of methods that are based on operator optimisation. Lastly, we consider the denoising by biharmonic inpainting to further investigate the question of data optimisation vs. operator optimisation.

5.6.1 Relation Between DbI and Homogeneous Diffusion

In Section 5.4.2 we derived a relation between the mask density d and the diffusion time T, given by $T = (1 - d^2)/(12d^2)$. To confirm that this relation allows for a good estimate of the diffusion time in practice, we perform an experiment on a 1-D signal, which is generated by extracting the 128th row of the *peppers* test image. Homogeneous diffusion is implemented using explicit Euler and the spatial discretisation from (5.49) with the number of iterations chosen such that the desired diffusion time T is reached. The result in Figure 5.6 demonstrates that the diffusion time obtained via Theorem 4 is a good approximation.

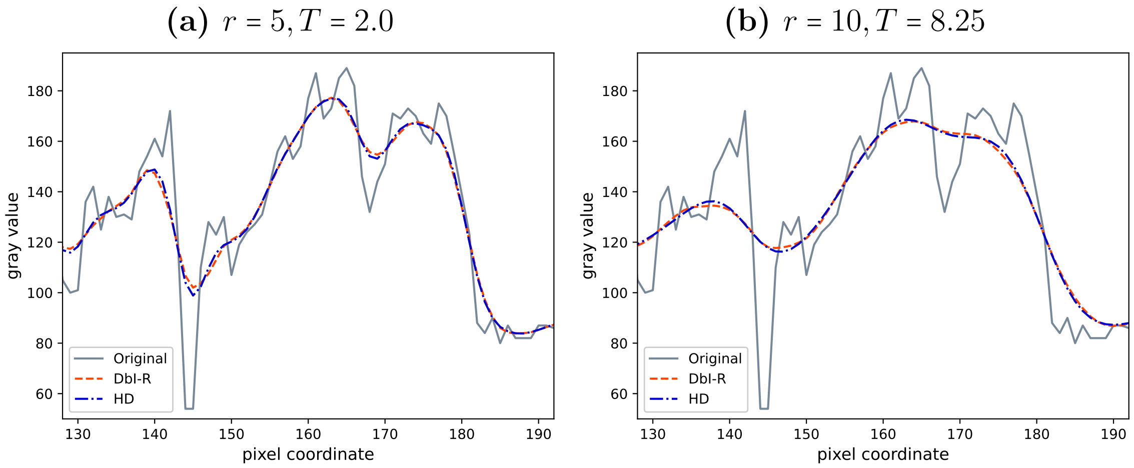


FIGURE 5.6: Comparison of denoising by inpainting with shifted regular masks (DbI-R) and homogeneous diffusion (HD) on a one-dimensional signal (128th row of the test image *peppers*). We display a section from the original signal and filtered versions obtained with denoising by inpainting with regular masks of spacing r and homogeneous diffusion filtering with diffusion time T, calculated according to Theorem 4. We see that both filters lead to very similar results, confirming that the approximation from the theorem is indeed realistic.

In Section 5.4.3 we extended this relation to 2D, yielding $T = (1-d^{\gamma})/(\beta d^{\gamma})$ with $\beta = 4.58$ and $\gamma = 1.3$. To confirm this, we now consider the 2-D *peppers* test image. We perform denoising by inpainting with 1024 randomly selected masks, as well as homogeneous diffusion filtering with the diffusion time calculated according to the above relation and compare the results. The experiments in Figure 5.7 visually and qualitatively confirm the accuracy of the relation in 2D.

5.6.2 Convergence

As we have shown in Section 5.3.1.4 the estimator converges to its expectation at a rate of $O(n^{-1/2})$ w.r.t. the RMSE. In Section 5.3.1.5 we introduced the idea of using low-discrepancy sequences. Theoretically, they should lead to much faster convergence, thus here we test whether this also holds in practice. In the experiments we again use the 256×256 test image *peppers*. We use two sampling strategies for the masks, whose sample means $\boldsymbol{c} = \frac{1}{n}\sum_{\ell=1}^{n} \boldsymbol{c}^{\ell}$ converge to the same expectation $\mathbb{E}[\boldsymbol{c}|\boldsymbol{f}]$. As a representative of a low-discrepancy sequence we use the R2 sequence [201] to create a sampling threshold in each pixel (see [201] for details). This leads to a more regular sampling pattern compared to using a purely random threshold. To make the experiment relevant to realistic scenarios, we use the analytic mask selection method from Section 5.5.2. We first test the mask convergence. To this end, we create $2^{16} = 65536$ masks via Poisson sampling

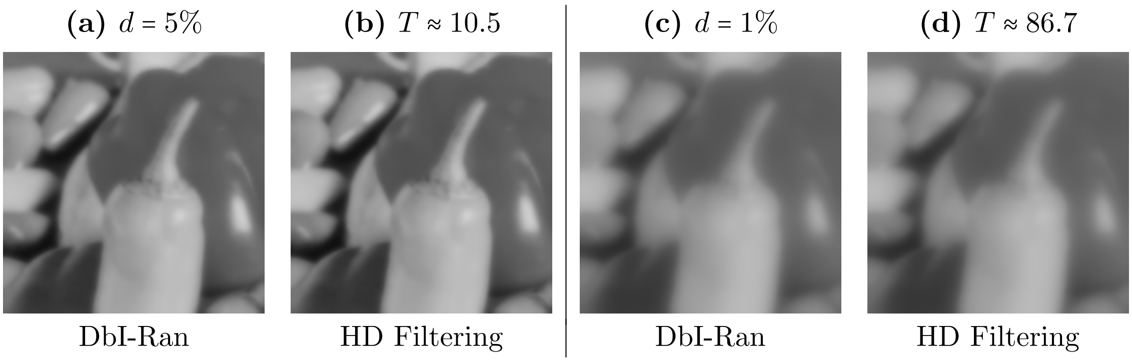
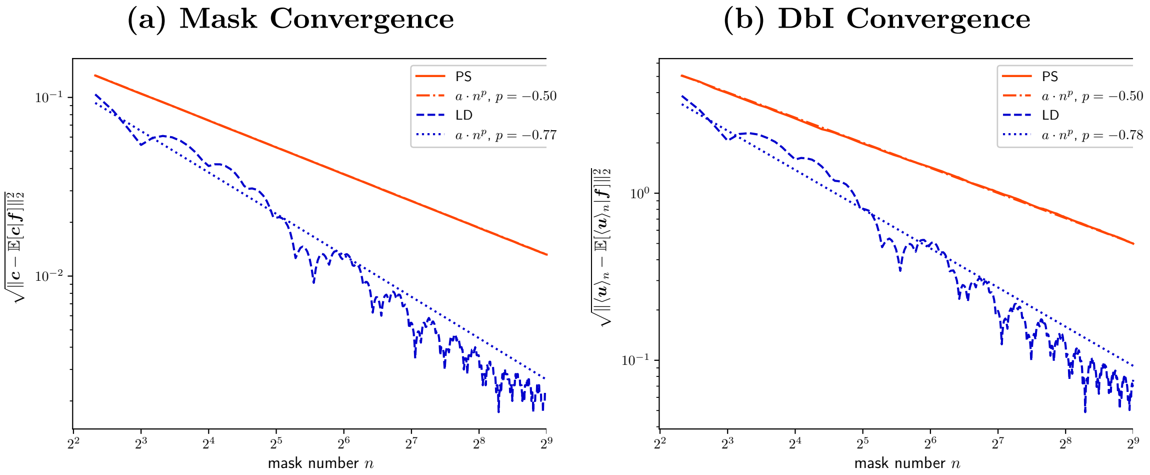


FIGURE 5.7: Comparison of denoising by inpainting with 1024 random masks (DbI-Ran) and homogeneous diffusion (HD) on the test image *peppers*. The diffusion times T corresponding to the mask densities d are calculated according to the result from Section 5.4.3. The MSE between (a) and (b) is 0.61 and the MSE between (c) and (d) is 6.37. This shows that the empirically derived relation is accurate, even for longer diffusion times.

and consider their average as converged to the expectation $\mathbb{E}[\boldsymbol{c}|\boldsymbol{f}]$. Then we sample masks with both sampling strategies and observe how the RMSE between sample mean and expectation evolves with n. Of course, we are more interested in the convergence of the DbI result $\langle \boldsymbol{u} \rangle_n$. Therefore, following a similar approach as for the masks, we create an individual "converged" DbI result for the two sampling methods, and again consider the RMSE between $\langle \boldsymbol{u} \rangle_n$ and the respective reference images. Figure 5.8 shows that the simple Poisson sampling leads to a convergence rate of $O(n^{-1/2})$ for the masks as well as for the DbI result, which is perfectly in line with the theory from Section 5.3.1.4. Through low-discrepancy sampling this rate approaches $O(n^{-1})$. By fitting a curve through the data, we get a convergence rate of $O(n^{-0.77})$ for the masks and $O(n^{-0.78})$ for the DbI result. The $O(n^{-1})$ estimate is typically achieved for low dimensions, so the difference of our results can be explained by the high dimensionality of our sampling problem. The experiments confirm that the sampling strategy based on low-discrepancy sequences is indeed able to improve the convergence in practice.

5.6.3 Data Optimisation for Denoising by Inpainting

In the next step, we investigate the edge-preserving filtering behaviour achieved by the use of adaptive masks. We first test the two spatial optimisation methods and compare the results to classical diffusion models. We show that DbI can yield results comparable to certain space-variant diffusion methods. Then we discuss the effect of tonal optimisation in the DbI setting. It should be noted that these experiments are meant to provide an illustration of the mask optimisation strategies and not to achieve the best denoising quality. As we have shown, these strategies can be applied in a more general setting than DbI with homogeneous diffusion inpainting. They are valid for the general probabilistic framework from Section 5.3.1, and as such they also extend to more complex operators (including nonlinear ones).


(a) Mask Convergence

(b) DbI Convergence

FIGURE 5.8: Convergence results for denoising by inpainting with the analytic method with Poisson sampling (PS) vs. low-discrepancy-based sampling (LD). (a) shows the convergence of the masks and (b) the convergence of the DbI result.

We perform experiments on the three standard test images *trui*, *peppers* and *walter* with a resolution of 256 × 256, that are corrupted with additive Gaussian noise with standard deviations $\sigma_n \in \{10, 20, 30\}$ that we do not clip. To ensure a fair comparison, we optimise the mask density and if required the pre- and post-smoothing parameter for the denoising by inpainting methods w.r.t. the MSE to the original image. We do this individually for each image and for each noise level using a grid search. In practice, these parameters need to be adapted to the noise level and the image content. We create 32 masks with each of the mask selection methods, except for the regular masks where the number is determined by the spacing and thus by the density. For the proposed probabilistic densification algorithm we set the number of candidate pixels per iteration to 16.

5.6.3.1 Spatial Optimisation

Firstly, we investigate the different spatial selection strategies proposed in Section 5.5 and compare the denoising results with the standard diffusion methods presented in Section 5.2.1. For the diffusion methods, which we discretise with an explicit scheme, we optimise the stopping time and if required the contrast parameter of the Charbonnier diffusivity [54].

As can be seen in Table 5.1, inpainting with regular masks leads to unsatisfying results, slightly worse than those obtained with homogeneous diffusion filtering. This is expected given the connections derived in Section 5.4.1. Note that the stopping time in homogeneous diffusion filtering can be tuned continuously, while the spacing of the regular mask can only be adapted in integer steps. The analytic method based on Poisson sampling of the smoothed Laplacian magnitude improves the results, especially at lower noise levels. Figure 5.9(c) shows how the mask pixels accumulate around important image structures, enabling an edge-preserving filtering behaviour. The densification method is able to further improve those results. The reason for this improvement can be seen in Figure 5.9(d). On top of selecting pixels at reasonable positions, the error in the mask is reduced drastically in comparison to the analytic method, because densification implicitly avoids noisy pixels. The adaptive mask selection strategies enable the denoising by inpainting method to produce results that are comparable to linear space-variant diffusion filtering. However, it cannot reach the quality of nonlinear diffusion. This is not surprising, as a feedback mechanism throughout the inpainting process is missing. Nonetheless, the results reveal that proper data optimisation enables DbI to compete with methods that optimise the operator, if they are of comparable complexity.

Although qualitatively the densification approach is better than the analytic method, its required run time is orders of magnitude larger, and this only gets worse for images of higher resolution. Due to the required number of inpaintings, the densification method takes about an hour to create a single mask with 10% density for our 256 × 256 pixel test images. In contrast, the analytic and the regular approaches allow instant mask generation in approximately a millisecond. Thus, the analytic method yields a reasonable spatial mask pixel distribution in a very short time and clearly has potential, if the error in the mask pixels can be reduced. We show next that this can be achieved by complementing the mask selection strategies with tonal optimisation.

Table 5.1: Results (MSE) for denoising by inpainting with regular masks, the densification method and the analytic method with 32 masks (fewer masks for the regular mask method). Comparison to classical diffusion-based denoising methods.

		trui			*peppers*			*walter*		
	noise level σ_n	10	20	30	10	20	30	10	20	30
DbI	regular	27.30	57.29	86.46	35.31	64.40	91.79	22.63	50.13	79.16
	densification	**19.34**	**42.72**	**68.01**	**24.36**	**47.27**	**69.89**	**13.40**	**29.65**	**47.65**
	analytic	21.49	49.71	79.79	25.14	51.70	79.91	16.41	37.83	62.08
Diff	homogeneous	24.12	50.18	76.12	32.16	59.77	84.58	19.65	42.76	66.87
	lin. space-var.	17.89	42.62	69.57	24.03	47.47	72.67	13.31	32.30	55.37
	nonlinear	**16.21**	**34.99**	**54.66**	**22.63**	**40.48**	**57.54**	**11.89**	**25.31**	**39.49**

5.6.3.2 Tonal Optimisation

As mentioned in Section 5.3.1.1, tonal optimisation leads to an MMSE estimate that is approximating instead of interpolating. If one assumes that mask pixels are erroneous due to the noise, this is certainly a desirable behaviour. We will evaluate its effect in the following. To this end, we apply tonal optimisation to the masks obtained by each of our spatial optimisation methods. We optimise the tonal values for each individual mask, before once again averaging the respective inpaintings to obtain the final denoised result.

The results in Table 5.2 reveal that the methods that do not consider the noise in the selection process get the greatest boost in performance. This confirms the conjecture that tonal optimisation is able to mitigate the negative effect of noisy mask pixels selection. We also observe that tonal optimisation decreases the error in the mask pixels for those methods. In Figure 5.9, the MSE at mask locations decreases from 405.09 to 271.80 for the regular mask, and from 410.63 to 306.62 for the analytic method. For probabilistic

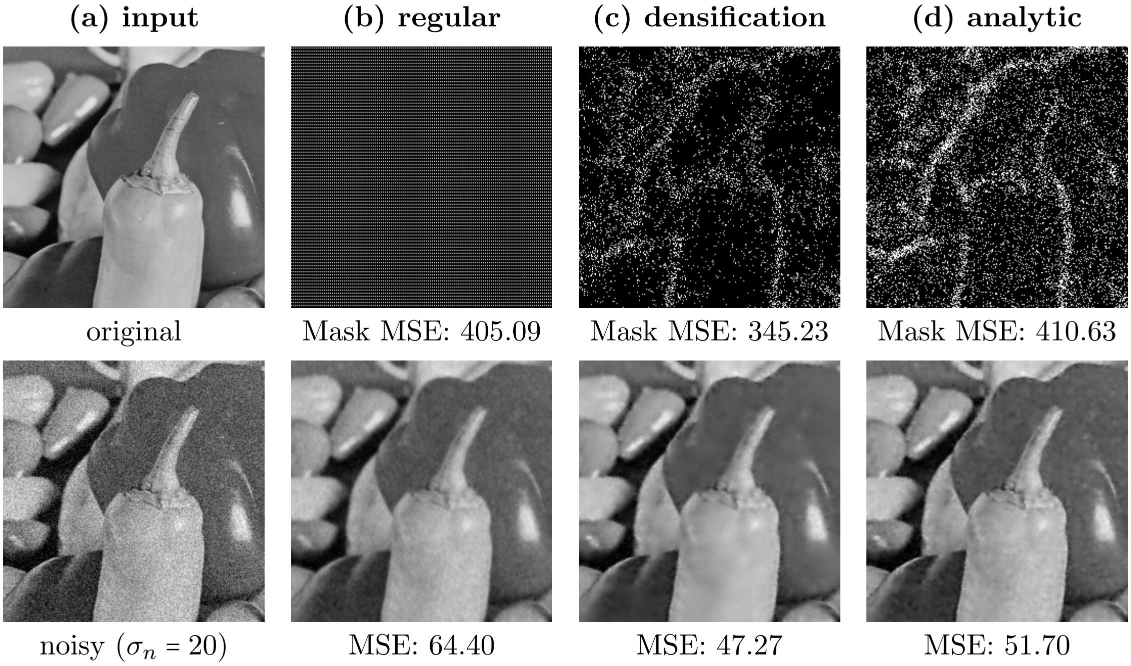


Figure 5.9: Results for denoising by inpainting with 32 masks (six masks for the regular mask method) for the different spatial optimisation methods on the test image *peppers* with $\sigma_n = 20$. Top row: (a) original image, (b)-(d) one representative out of all the masks for every method. The MSE is computed at mask pixels. Bottom row: (a) noisy image, (b)-(d) denoising by inpainting results with optimised parameters and the MSE in the entire image. We see that our analytic method and the densification method adapt the mask point locations to the structure of the image. Densification additionally avoids choosing noisy mask pixels, leading to a smaller error in the mask pixels and eventually to a better reconstruction.

densification, tonal optimisation barely changes the final results, as well as the mask MSE (which even increases slightly from 345.23 to 356.12 in the example).

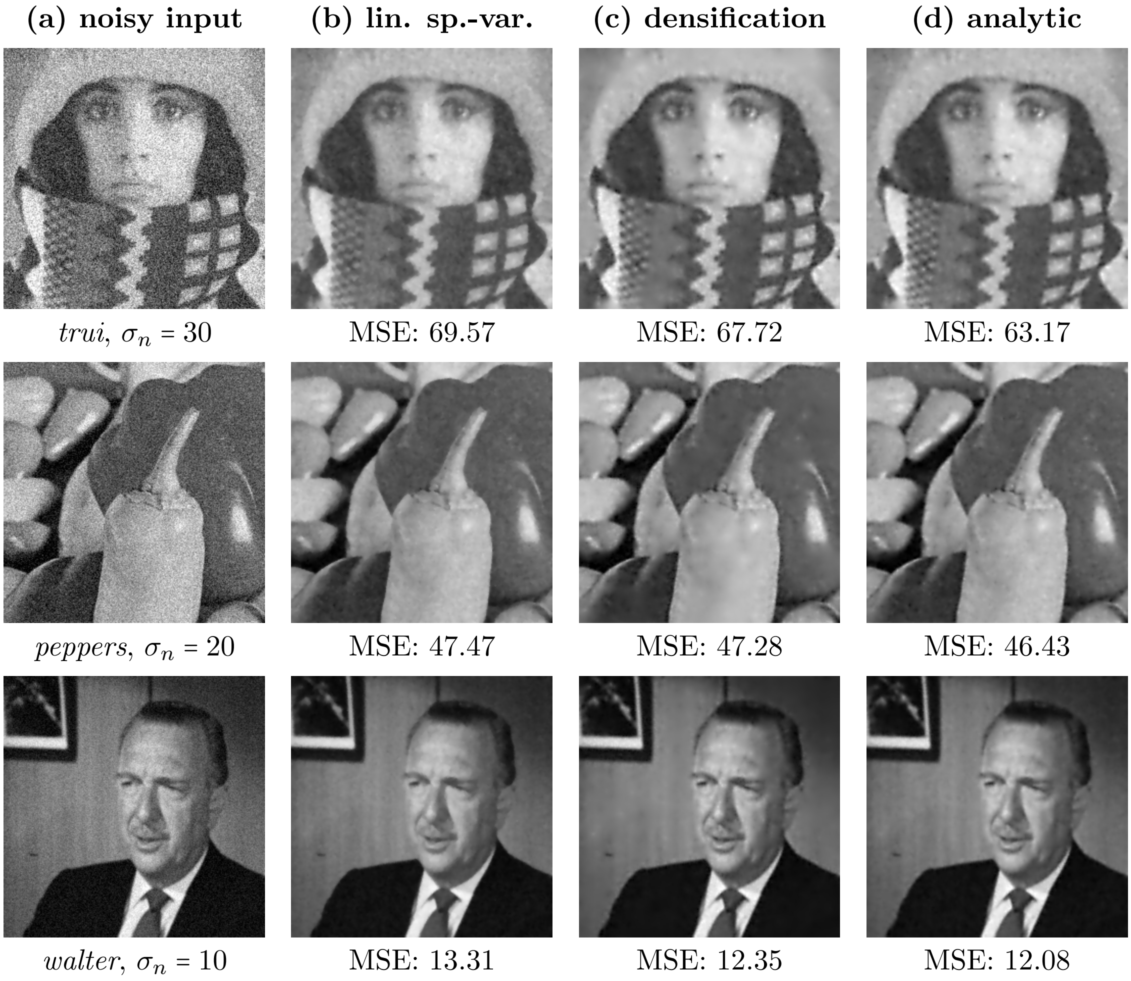
We see that tonal optimisation enables the analytic method to produce results of quality comparable to those of the densification method, and of better quality than space-variant diffusion. Although the tonal optimisation step takes some additional seconds, the analytic method is still orders of magnitude faster than the densification method. Figure 5.10 shows a selection of resulting images comparing the two adaptive mask selection methods with tonal optimisation and linear space-variant diffusion, as the diffusion method that leads to the most similar results.

5.6.4 Denoising by Biharmonic Inpainting

Our previous results reveal that optimising the data instead of the operator constitutes an interesting alternative for image denoising. To further substantiate this idea, we now adapt the inpainting operator within the DbI framework. We consider biharmonic inpainting as a representative of a higher-order polyharmonic operator.

Table 5.2: Results (MSE) for denoising by inpainting with regular masks, the densification method and the analytic method with 32 masks (less masks for the regular mask method) including tonal optimisation. Comparison to classical diffusion-based denoising methods.

		trui			*peppers*			*walter*		
	noise level σ_n	10	20	30	10	20	30	10	20	30
DbI	regular	22.32	48.77	76.06	32.65	60.92	87.05	16.36	39.54	64.45
	densification	18.46	41.56	67.72	24.42	47.28	70.21	12.35	28.13	45.92
	analytic	**17.24**	**39.49**	**63.17**	**23.68**	**46.43**	**68.55**	**12.08**	**27.66**	**45.36**
Diff	homogeneous	24.12	50.18	76.12	32.16	59.77	84.58	19.65	42.76	66.87
	lin. space-var.	17.89	42.62	69.57	24.03	47.47	72.67	13.31	32.30	55.37
	nonlinear	**16.21**	**34.99**	**54.66**	**22.63**	**40.48**	**57.54**	**11.89**	**25.31**	**39.49**



trui, $\sigma_n = 30$ MSE: 69.57 MSE: 67.72 MSE: 63.17

peppers, $\sigma_n = 20$ MSE: 47.47 MSE: 47.28 MSE: 46.43

walter, $\sigma_n = 10$ MSE: 13.31 MSE: 12.35 MSE: 12.08

Figure 5.10: Visual comparison of linear space-variant diffusion and denoising by inpainting with the densification method and the analytic method on three test images with noise. Both DbI methods are using tonal optimisation.

It has been shown that the biharmonic operator can have quality advantages over homogeneous diffusion (i.e., the harmonic operator) in classical sparse inpainting [57, 89, 209]. Biharmonic inpainting is given by the PDE

$$\big(c(\boldsymbol{x}) + (1 - c(\boldsymbol{x}))\Delta^2\big)u(\boldsymbol{x}) = c(\boldsymbol{x})f(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \in \Omega, \tag{5.67}$$

with $\Delta^2 u = \partial_{xxxx}u + 2\partial_{xxyy}u + \partial_{yyyy}u$ and reflecting boundary conditions $\partial_{\boldsymbol{n}}u(\boldsymbol{x}) = 0$ and $\partial_{\boldsymbol{n}}\Delta u(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \in \partial\Omega$. It can be derived from the following variational formulation (analogously to (5.12)):

$$\min_u \int_\Omega (\Delta u(\boldsymbol{x}))^2 \, d\boldsymbol{x}, \text{ such that } u(\boldsymbol{x}) = f(\boldsymbol{x}) \text{ for } \boldsymbol{x} \in K. \tag{5.68}$$

This shows that biharmonic inpainting penalises second-order derivatives. Biharmonic inpainting does not suffer from the typical singularities at mask points that homogeneous diffusion inpainting produces. On the other hand it can produce over- and undershoots, since it does not guarantee a maximum-minimum principle. We evaluate the potential of biharmonic inpainting for denoising by comparing it to homogeneous diffusion inpainting. To ensure that the results reflect the quality of the operators, we first perform the experiment on fully random masks.

Our results in Table 5.3 show that biharmonic inpainting does lead to an improvement, and it is largest at low noise levels. This is to be expected, as the method is not as radical as homogeneous diffusion inpainting, since it penalises second degree instead of first degree derivatives. However, already tonal optimisation as a first data optimisation step neutralises this advantage and the two methods perform similarly. These results support our reasoning that data optimisation plays a significant role for the denoising abilities of our framework, being more important than the use of more complex, higher-order models. Further experiments on spatially optimised masks (see Table 5.4) confirm our findings, and even shift the advantage towards homogeneous diffusion inpainting. When comparing to previous results from classical sparse image inpainting, one has to consider that the singularities, that homogeneous diffusion inpainting suffers from, are suppressed by the averaging in the DbI framework. Thus, this disadvantage of homogeneous diffusion inpainting does not come into play in our scenario. Lastly, one should keep in mind that biharmonic inpainting leads to a higher condition number of the inpainting matrix, and consequently each inpainting is numerically more burdensome and less efficient.

Table 5.3: Results (MSE) for denoising by inpainting with 32 random masks using homogeneous diffusion (HD) and biharmonic (BI) inpainting, without and with tonal optimisation (TO).

	trui			*peppers*			*walter*		
noise level σ_n	10	20	30	10	20	30	10	20	30
HD, without TO	30.53	65.51	100.51	36.01	71.20	104.21	26.93	60.21	94.37
BI, without TO	**24.23**	**56.37**	**93.51**	**33.28**	**66.61**	**102.49**	**19.16**	**48.19**	**82.92**
HD, with TO	23.10	49.83	**76.57**	**31.98**	**59.75**	**85.28**	18.09	41.79	66.40
BI, with TO	**22.21**	**49.74**	77.25	33.27	61.84	87.92	**16.52**	**39.53**	**65.03**

Table 5.4: Results (MSE) for denoising by inpainting with 32 masks obtained with the analytic method using homogeneous diffusion (HD) and biharmonic (BI) inpainting, without and with tonal optimisation (TO).

	trui			*peppers*			*walter*		
noise level σ_n	10	20	30	10	20	30	10	20	30
HD, without TO	21.49	49.71	**79.79**	**25.14**	**51.70**	**79.91**	16.41	37.83	**62.08**
BI, without TO	**19.01**	**47.47**	82.39	25.83	55.42	90.28	**14.16**	**37.15**	68.25
HD, with TO	17.24	**39.49**	**63.17**	**23.68**	**46.43**	**68.55**	12.08	27.66	**45.36**
BI, with TO	**17.18**	40.45	66.13	25.35	49.27	72.68	**11.74**	**27.22**	45.70

5.7 Conclusions

Our work is the first that links the tasks of PDE-based image inpainting and denoising in a systematic way, by providing an explicit connection between homogeneous diffusion inpainting and denoising through a relation between the diffusion time and the mask density. Our *denoising by inpainting (DbI)* framework achieves denoising by averaging inpainting results with different sparse masks of the same density. It constitutes a means to investigate the connections between PDE-based denoising and inpainting and allows us to evaluate the denoising potential of PDE-based inpainting methods. We have established a probabilistic theory with convergence estimates for the framework, and have extended it to a deterministic version by the use of low-discrepancy sequences. We have further shown that this framework computes an approximation to an MMSE estimate. For non-adaptive masks we have linked the framework to classical diffusion via a one-to-one relationships between the mask density and the diffusion time. We have demonstrated that a simple operator can exhibit space-variant filtering behaviour, when supplemented with adaptive data selection strategies. Experiments with a higher-order inpainting operator, which can be more powerful than homogeneous diffusion inpainting [57, 89, 209], have underlined the importance of choosing appropriate data over more complex operators. For data optimisation specific to denoising by inpainting, we have presented two distinct, fundamental strategies. The densification method from the conference paper [2] aims at finding pixels that represent the data well. Thereby, it implicitly avoids the selection of noisy mask pixels during spatial optimisation. On the contrary, we have proposed a new approach, where the selection of noisy pixels is tolerated in the spatial optimisation but is compensated for by the tonal optimisation.

Our work constitutes an unconventional, new viewpoint on image denoising: By using a simple inpainting operator but focusing on adequate data selection we *shift the priority from optimising the filter model to optimising the considered data.* Moreover, our densification strategy allows us to find the most trustworthy pixels in the data. This shows that *simple filter operators such as homogeneous diffusion can give deep insights into data.* Last but not least, we have seen that the filling-in effect is not only useful in variational optic flow models and in PDE-based inpainting, but also in denoising. This emphasises its fundamental role in digital image analysis, which is in full agreement with classical results from biological vision [245].

While our focus in the present chapter was on gaining fundamental insights into the potential of inpainting ideas for denoising, our future work will deal with various modifications to make these ideas also applicable to more recent denoising methods. To this end, we are going to consider more sophisticated inpainting operators [209] and data selection strategies [68], including neural ones [192], and the incorporation of more advanced types of data [120]. Such future work should also extend our theory to, e.g., space-variant and nonlinear operators.

Chapter 6

Perceptual Error Optimisation for Monte Carlo Rendering

6.1 Introduction

Monte Carlo sampling produces approximation error. In rendering, this error can cause visually displeasing image artifacts, unless control is exerted over the correlation of the individual pixel estimates. A standard approach is to decorrelate these estimates by randomising the samples independently for every pixel, turning potential structured artifacts into white noise.

In digital halftoning, the error induced by quantising continuous-tone images has been studied extensively. Such studies have shown that a blue-noise distribution of the quantisation error is perceptually optimal [231], achieving substantially higher image fidelity than a white-noise distribution. Recent works have proposed empirical means to transfer these ideas to image synthesis [5, 93, 103, 104]. Instead of randomising the pixel estimates, these methods introduce *negative* correlation between neighbouring pixels, exploiting the local smoothness in images to push the estimation error to the high-frequency spectral range.

We propose a theoretical formulation of perceptual error for image synthesis which unifies prior methods in a common framework and formally justifies the desire for blue-noise error distribution. We extend the comparatively simpler problem of digital halftoning [146] where the ground-truth image is given, to the substantially more complex one of rendering where the ground truth is the sought result and thus unavailable. Our formulation bridges the gap between multi-tone halftoning and rendering by interpreting Monte Carlo estimates for a pixel as its admissible 'quantisation levels'. This insight allows virtually any halftoning method to be adapted to rendering. We demonstrate this for the three main classes of halftoning algorithms: dither-mask halftoning, error diffusion halftoning, and iterative energy minimisation halftoning.

Existing methods [93, 103, 104] can be seen as variants of dither-mask halftoning. They distribute pixel error according to masks that are optimised w.r.t. a target kernel, typically a Gaussian. The kernel can be interpreted as an approximation to the human visual system's point spread function [66, 183]. We revisit the kernel-based perceptual model from halftoning [11, 183, 223] and adapt it to rendering. The resulting energy can be directly used for optimising Monte Carlo error distribution without the need for a mask. This formulation helps us expose the underlying assumptions of existing methods and quantify their limitations. In summary:

- We formulate an optimisation problem for rendering error by leveraging kernel-based perceptual models from halftoning.
- Our formulation unifies prior blue-noise error distribution methods and makes all their assumptions explicit, outlining general guidelines for devising new methods in a principled manner.
- Unlike prior methods, our formulation simultaneously optimises for both the magnitude and the image distribution of pixel error.
- We devise four different practical algorithms based on iterative minimisation, error diffusion, and dithering from halftoning.
- We demonstrate substantial visual improvements over prior art, while using the same input rendering data.

6.1.1 Outline

We review the related work in Section 6.2. We then present the perceptual error model used throughout this chapter in Section 6.3. In Section 6.4 we discuss the optimisation problem resulting from our formulation. We consider practical applications of the framework in Section 6.5. In Section 6.6 we discuss various extensions that are relevant to rendering. In Section 6.7 we provide a multitude of experiments that illustrate the power of our formulation. We discuss some additional aspects in Section 6.8, and then conclude the chapter in Section 6.9.

6.2 Related Work

Our work focuses on reducing and optimising the distribution of Monte Carlo pixel-estimation error. In this section we review prior work with similar goals in digital halftoning (Section 6.2.1) and image synthesis guided by energy-based (Section 6.2.2) and perception-based (Section 6.2.3) error metrics. We achieve error reduction through careful sample placement and processing, and discuss related rendering approaches (Section 6.2.4).

6.2.1 Digital Halftoning

Digital halftoning [146] involves creating the illusion of continuous-tone images through the arrangement of binary elements; various algorithms target different display devices.

Bayer [23] developed the widely used dispersed-dot ordered-dither patterns. Allebach and Liu [8] introduced the use of randomness in clustered-dot ordered dithering. Ulichney [231] introduced *blue-noise* patterns that yield better perceptual quality, and Mitsa and Parker [170] mimicked those patterns to produce dither arrays (i.e., masks) with high-frequency characteristics. Sullivan et al. [223] developed a Fourier-domain energy function to obtain visually optimal halftone patterns; the optimality is defined w.r.t. computational models of the human visual system. Analoui and Allebach [11] devised a practical algorithm for blue-noise dithering through a spatial-domain interpretation of Sullivan et al.'s model. Their approach was later refined by Pappas and Neuhoff [183].

The void-and-cluster algorithm [232] uses a Gaussian kernel to create dither masks with isotropic blue-noise distribution. This approach has motivated various structure-aware halftoning algorithms in graphics [53, 179, 182]. In the present work, we leverage the kernel-based model [11, 183] in the context of Monte Carlo rendering [124].

6.2.2 Quantitative Error Assessment in Rendering

It is convenient to measure the error of a rendered image as a single value; vector norms like the mean squared error (MSE) are most commonly used. However, it is widely acknowledged that such simple metrics do not accurately reflect visual quality as they ignore the perceptually important spatial arrangement of pixels. Various theoretical frameworks have been developed in the spatial [140, 175] and Fourier [215] domains to understand the error reported through these metrics. The error spectrum ensemble [48] measures the frequency-space distribution of the error.

Many denoising methods [253] employ the aforementioned metrics to obtain noise-free results from noisy renderings. Even if the most advanced denoising techniques driven by such metrics can efficiently steer adaptive sampling [49, 130, 142], they locally determine the number of samples per pixel, ignoring the aspect of their specific layout in screen space.

Our optimisation framework employs a perceptual MSE-based metric that accounts for both the magnitude and the spatial distribution of pixel-estimation error. We argue that the spatial sample layout plays a crucial role in the perception of a rendered image; the most commonly used error metrics do not capture this aspect.

6.2.3 Perceptual Error Assessment in Rendering

The study of the human visual system (HVS) is still ongoing, and well understood are mostly the early stages of the visual pathways from the eye optics, through the retina, to the visual cortex. This limits the scope of existing HVS computational models used in imaging and graphics. Such models should additionally be computationally efficient and generalise over the simplistic stimuli that have been used in their derivation through psychophysical experiments.

Contrast Sensitivity Function. The contrast sensitivity function (CSF) is one of the core HVS models that fulfils the above conditions and comprehensively characterises overall optical [69, 247] and neural [217] processes in detecting contrast visibility as a function of spatial frequency. While originally modelled as a band-pass filter [22, 67], the CSF's shape changes towards a low-pass filter with retinal eccentricity [184, 202] and reduced luminance adaptation in scotopic and mesopic levels [249]. Low-pass characteristics are also inherent for chromatic CSFs [34, 171, 249]. In many practical imaging applications, e.g., JPEG compression [199], rendering [197], or halftoning [183], the CSF is modelled as a low-pass filter, which also allows for better control of image intensity. By normalising such a CSF by the maximum contrast-sensitivity value, a unit-less function akin to the modulation transfer function (MTF) can be derived [66, 159, 160, 217, 223] that after transforming from the frequency to the spatial domain results in the point spread function (PSF) [11, 183]. Following Pappas and Neuhoff [183], we approximate such a PSF by a Gaussian filter; the resulting error is practically negligible for a pixel density of 300 dots per inch (dpi) and observer-to-screen distance larger than 60 cm.

Advanced Quality Metrics. More costly, and often less robust, modelling of the HVS beyond the CSF is performed in advanced quality metrics [67, 154, 161]. Such metrics have been adapted to rendering to guide the computation to image regions where the visual error is most strongly perceived [33, 34, 84, 172, 197, 235]. An important application is visible noise reduction in path tracing via content-adaptive sample-density control [33, 34, 197]. Our framework enables significant reduction of noise visibility for the same sampling budget.

6.2.4 Blue-noise Error Distribution in Rendering

Mitchell [169] first observed that high-frequency error distribution is desirable for stochastic rendering. Only recently, Georgiev and Fajardo [93] adopted techniques from halftoning to correlate pixel samples in screen space and distribute path-tracing error as blue noise, with substantial perceptual quality improvements. Heitz et al. [104] built on this idea to develop a progressive quasi-Monte Carlo sampler that further improves quality. Ahmed and Wonka [5] proposed a technique to coordinate quasi-Monte Carlo samples in screen space inspired by error diffusion.

Motivated by the results of Georgiev and Fajardo [93], Heitz and Belcour [103] devised a method to directly optimise the distribution of pixel estimates, without operating on individual samples. Their pixel permutation strategy fits the initially white-noise pixel intensities to a prescribed blue-noise mask. This approach scales well with sample count and dimension, though its reliance on prior pixel estimates makes it practical only for animation rendering where it is susceptible to quality degradation.

We propose a perceptual error framework that unifies these two general approaches, exposing the assumptions of existing methods and providing guidelines to alleviate some of their drawbacks.

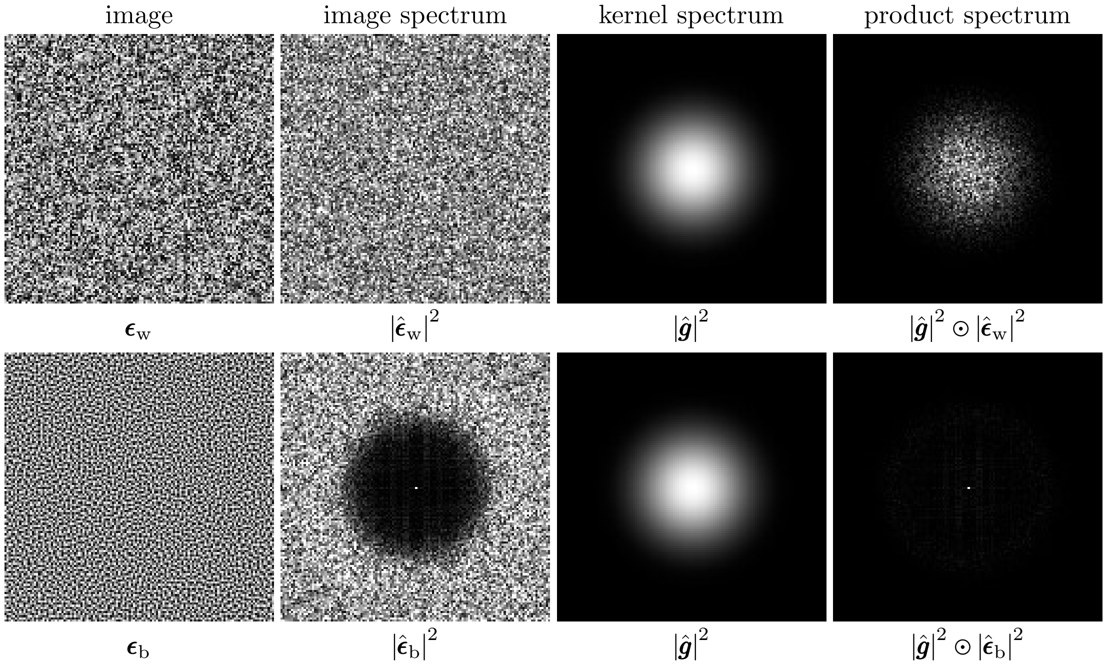


FIGURE 6.1: Error images $\boldsymbol{\epsilon}_{\mathrm{w}}$ and $\boldsymbol{\epsilon}_{\mathrm{b}}$ with respective white-noise, $|\hat{\boldsymbol{\epsilon}}_{\mathrm{w}}|^2$, and blue-noise, $|\hat{\boldsymbol{\epsilon}}_{\mathrm{b}}|^2$, power spectra. For a low-pass kernel $\boldsymbol{g}$ modelling the PSF of the HVS (here a Gaussian with std. dev. $\sigma = 1$), the product of its spectrum $|\hat{\boldsymbol{g}}|^2$ with $|\hat{\boldsymbol{\epsilon}}_{\mathrm{b}}|^2$ has lower magnitude than the product with $|\hat{\boldsymbol{\epsilon}}_{\mathrm{w}}|^2$. This corresponds to lower perceptual sensitivity to $\boldsymbol{\epsilon}_{\mathrm{b}}$, even though $\boldsymbol{\epsilon}_{\mathrm{w}}$ has the same amplitude as it is obtained by randomly permuting the pixels of $\boldsymbol{\epsilon}_{\mathrm{b}}$.

6.3 Perceptual Error Model

Our aim is to produce Monte Carlo renderings that, at a fixed sampling rate, are perceptually as close to the ground truth as possible. This goal requires formalising the perceptual image error along with an optimisation problem that minimises it. In this section, we build a perceptual model upon the extensive studies done in the halftoning literature. We will discuss how to efficiently solve the resulting optimisation problem in Section 6.4.

Given a ground-truth image $\boldsymbol{I}$ and its quantised or stochastic approximation $\boldsymbol{Q}$, we denote the (signed) error image by

$$\boldsymbol{\epsilon} = \boldsymbol{Q} - \boldsymbol{I}. \tag{6.1}$$

To minimise the error, it is convenient to quantify it as a single value. A common approach is to take the $\mathcal{L}_1$, $\mathcal{L}_2$, or $\mathcal{L}_\infty$ norm of the image $\boldsymbol{\epsilon}$ interpreted as a vector. Such simple metrics are permutation-invariant, i.e., they account for the *magnitudes* of individual pixel errors but not for their *distribution* over the image. This distribution is an important factor for the perceived fidelity, since contrast perception is an inherently spatial characteristic of the HVS (Section 6.2.3). Our model is based on perceptual halftoning metrics that capture both the magnitude and the distribution of error.

6.3.1 Motivation

Halftoning metrics model the processing done by the HVS as a convolution of the error image $\boldsymbol{\epsilon}$ with a kernel $\boldsymbol{g}$:

$$E = \|\boldsymbol{g} * \boldsymbol{\epsilon}\|_2^2 = \|\hat{\boldsymbol{g}} \odot \hat{\boldsymbol{\epsilon}}\|_2^2 = \langle |\hat{\boldsymbol{g}}|^2, |\hat{\boldsymbol{\epsilon}}|^2 \rangle. \tag{6.2}$$

The convolution is equivalent to the element-wise product of the corresponding Fourier spectra $\hat{\boldsymbol{g}}$ and $\hat{\boldsymbol{\epsilon}}$, whose 2-norm in turn equals the inner product of the power spectra images $|\hat{\boldsymbol{g}}|^2$ and $|\hat{\boldsymbol{\epsilon}}|^2$. Sullivan et al. [223] optimised the error image $\boldsymbol{\epsilon}$ to minimise the error (6.2) w.r.t. a kernel $\boldsymbol{g}$ that approximates the HVS's modulation transfer function $|\hat{\boldsymbol{g}}|$ (MTF) [66]. Analoui and Allebach [11] used a similar model in the spatial domain with a kernel that approximates the PSF[1] of the human eye. That kernel is low-pass, and the optimisation naturally yields blue-noise[2] distribution in the error image [11], as we show later in Fig. 6.6. The blue-noise distribution can thus be seen as by-product of the optimisation which pushes the spectral components of the error to the frequencies least visible to the human eye (see Fig. 6.1).

To better understand the spatial aspects of contrast sensitivity in the HVS, the MTF is usually modelled over a range of viewing distances [67]. This is done to account for the fact that with increasing viewer distance, spatial frequencies in the image are projected to higher spatial frequencies onto the retina. These frequencies eventually become invisible, filtered out by the PSF which expands its corresponding kernel in image space. We recreate this experiment to see the impact of distance on the image error. In Fig. 6.2, we convolve white- and blue-noise distributions with a Gaussian kernel of increasing standard deviation corresponding to increasing observer-to-screen distance. The high-frequency blue-noise distribution reaches a homogeneous state (where the tone appears constant) faster compared to the all-frequency white noise. This means that high-frequency error becomes indiscernible at closer viewing distances, where the HVS ideally has not yet started filtering out actual image detail which is typically low- to mid-frequency. In Section 6.6 we discuss how the kernel's standard deviation encodes the viewing distance w.r.t. to the screen resolution.

6.3.2 Our Model

In rendering, the value of each pixel i is a light-transport integral. Point-sampling its integrand with a sample set S yields a pixel estimate $Q_i(S)$. The signed pixel error is thus a function of the sample set: $\epsilon_i(S) = Q_i(S) - I_i$, where I_i is the reference (i.e., ground-truth) pixel value. The error of the entire image can be written as

$$\boldsymbol{\epsilon}(\boldsymbol{S}) = \boldsymbol{Q}(\boldsymbol{S}) - \boldsymbol{I}, \tag{6.3}$$

where $\boldsymbol{S} = \{S_1, \ldots, S_N\}$ is an "image" containing the sample set for all N pixels. With these definitions, we can express the perceptual error in Eq. (6.2) for the case of Monte

[1] The MTF is the magnitude of the Fourier transform of the PSF.

[2] The term "blue noise" is often used loosely to refer to any isotropic spectrum with minimal low-frequency content and no concentrated energy spikes.

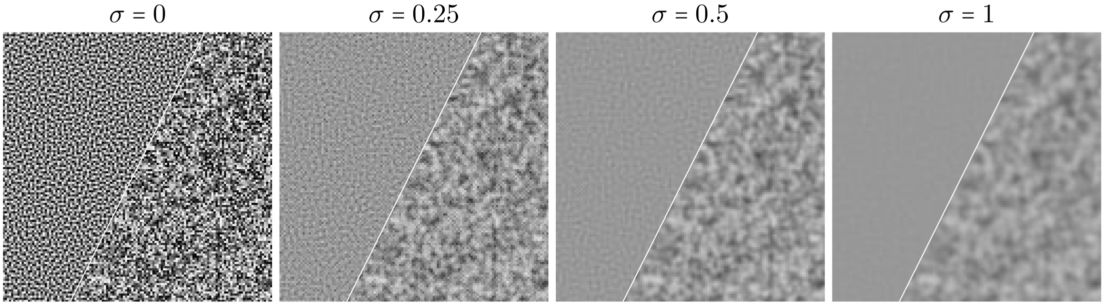


Figure 6.2: The appearance of blue noise (left images) converges to a constant image faster than white noise (right images) with increasing observer distance, here emulated via the standard deviation σ of a Gaussian kernel. We provide a formal connection between σ and the viewing distance in Section 6.6.

Carlo rendering as a function of the sample-set image $\boldsymbol{S}$, given a kernel $\boldsymbol{g}$:

$$E(\boldsymbol{S}) = \|\boldsymbol{g} * \boldsymbol{\epsilon}(\boldsymbol{S})\|_2^2. \tag{6.4}$$

Our goal is to minimise the perceptual error (6.4). We formulate this task as an optimisation problem:

$$\boxed{\min_{\boldsymbol{S}\in\boldsymbol{\Omega}} E(\boldsymbol{S}) = \min_{\boldsymbol{S}\in\boldsymbol{\Omega}} \|\boldsymbol{g} * (\boldsymbol{Q}(\boldsymbol{S}) - \boldsymbol{I})\|_2^2. \qquad (6.5)}$$

The minimising sample-set image $\boldsymbol{S}$ yields an image estimate $\boldsymbol{Q}(\boldsymbol{S})$ that is closest to the reference $\boldsymbol{I}$ w.r.t. the kernel $\boldsymbol{g}$. The search space $\boldsymbol{\Omega}$ is the set of all possible locations for every sample of every pixel. The total number of samples in $\boldsymbol{S}$ is typically bounded by a given target sampling budget. Practical considerations may also restrict the search space $\boldsymbol{\Omega}$, as we will exemplify in the following section.

Note that the classical MSE metric corresponds to using a zero-width (i.e., one-pixel) kernel $\boldsymbol{g}$ in Eq. (6.4). However, the MSE accounts only for the magnitude of the error $\boldsymbol{\epsilon}$, while using wider kernels (such as the PSF) accounts for both magnitude and distribution. Consequently, while the MSE can be minimised by optimising pixels independently, minimising the perceptual error requires coordination between pixels. In the following section, we devise strategies for solving this optimisation problem.

6.4 Discrete Optimisation

In our optimisation problem (6.5), the search space for each sample in every pixel is a high-dimensional unit hypercube. Every point in this so-called primary sample space maps to a light-transport path in the scene [193]. Optimising for the sample-set image $\boldsymbol{S}$ thus entails evaluating the contributions $\boldsymbol{Q}(\boldsymbol{S})$ of all corresponding paths. This evaluation is costly, and for any non-trivial scene, $\boldsymbol{Q}$ is a function with complex shape and many discontinuities. This precludes us from studying all (uncountably infinite) sample locations in practice.

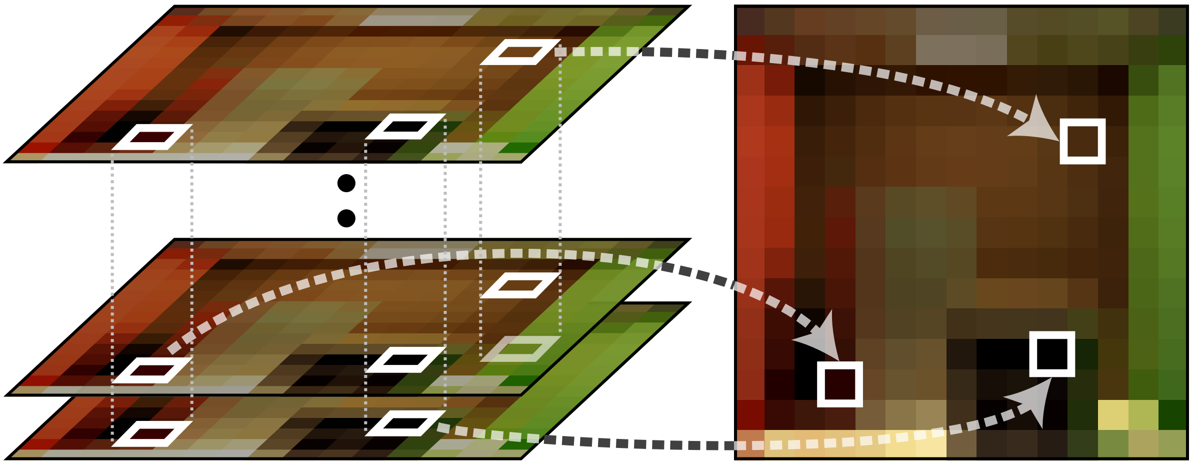


Figure 6.3: Vertical search space optimisation.

To make the problem tractable, we restrict the search in each pixel to a finite number of (pre-defined) sample sets. We devise two variants of the resulting discrete optimisation problem, which differ in their definition of the global search space $\boldsymbol{\Omega}$. In the first variant, each pixel has a separate list of sample sets to choose from ("vertical" search space). The setting is similar to that of (multi-tone) halftoning [146], which allows us to import classical optimisation techniques from that field, such as iterative minimisation, error diffusion, and mask-based dithering. In the second variant, each pixel has one associated sample set, and the search space comprises permutations of these assignments ("horizontal" search space). We develop a greedy iterative optimisation method for this second variant.

In contrast to halftoning, in our setting the ground-truth image $\boldsymbol{I}$—required to compute the error image $\boldsymbol{\epsilon}$ during optimisation—is not readily available. Below we describe our algorithms assuming the ground truth is available; in Section 6.5 we will discuss how to substitute it with a surrogate to make the algorithms practical.

6.4.1 Vertical Search Space

Our first variant considers a "vertical" search space where the sample set for each of the N image pixels is one of M given sets:[3]

$$\boldsymbol{\Omega} = \{\boldsymbol{S} = \{S_1, \ldots, S_N\} : S_i \in \{S_{i,1}, \ldots, S_{i,M}\}\}. \tag{6.6}$$

The objective is to find a sample set S_i for every pixel i such that all resulting pixel estimates together minimise the perceptual error (6.4). This is equivalent to directly optimising over the M possible estimates $Q_{i,1}, \ldots, Q_{i,M}$ for each pixel, with $Q_{i,j} = Q_i(S_{i,j})$. These estimates can be obtained by pre-rendering a stack of M images

[3] For notational simplicity, and without loss of generality, we assume that the number of candidate sample sets M is the same for all pixels; in practice can vary per pixel.

$\boldsymbol{Q}_j = \{Q_{1,j}, \ldots, Q_{N,j}\}$, for $j = 1..M$. The resulting minimisation problem reads:

$$\min_{\boldsymbol{O}: O_i \in \{Q_{i,1}, \ldots, Q_{i,M}\}} \|\boldsymbol{g} * (\boldsymbol{O} - \boldsymbol{I})\|_2^2. \tag{6.7}$$

This problem is almost identical to that of multi-tone halftoning. The difference is that in our setting the "quantisation levels", i.e., the pixel estimates, are distributed non-uniformly and vary per pixel as they are not fixed but are the result of point-sampling a light-transport integral. This similarity allows us to directly apply existing optimisation techniques from halftoning. We consider three such methods, which we outline in Alg. 1 and describe next.

Iterative minimisation. State-of-the-art halftoning methods attack the problem (6.7) directly via greedy iterative minimisation [11, 183]. After initialising every pixel to a random quantisation level, we traverse the image in serpentine order (as is standard practice in halftoning) and for each pixel choose the level that minimises the energy. Several full-image iterations are performed; in our experiments convergence to a local minimum is achieved within 10–20 iterations.

As a further improvement, the optimisation can be terminated when no pixels are updated within one full iteration, or when the perceptual-error reduction rate drops below a certain threshold. Traversing the pixels in random order allows terminating at any point but converges slightly slower.

Error diffusion. A classical halftoning algorithm, error diffusion scans the image pixel by pixel, snapping each reference value to the closest quantisation level and distributing the resulting pixel error to yet-unprocessed nearby pixels according to a given kernel $\boldsymbol{\kappa}$. We use the empirically derived kernel of Floyd and Steinberg [85] which has been shown to produce an output that approximately minimises Eq. (6.7) [107]. Error diffusion is faster than iterative minimisation but yields less optimal solutions.

Dithering. The fastest halftoning approach quantises pixel values using thresholds stored in a pre-computed dither mask (or matrix) [218]. For each pixel, the two quantisation levels that tightly envelop the reference value (in terms of brightness) are found, and one of the two is chosen based on the threshold assigned to the pixel by the mask.

Dithering can be understood as performing the perceptual error minimisation in two steps. First, an offline optimisation encodes the error distribution optimal for the target kernel $\boldsymbol{g}$ into a mask. Then, for a given image, the error magnitude is minimised by restricting the quantisation to the two closest levels per pixel, and the mask-driven choice between them applies the target distribution of error.

ALGORITHM 1: Three algorithms to (approximately) solve the vertical search space optimisation problem (6.7). The output is an image $\boldsymbol{O} = \{O_1, \dots, O_N\}$, given a reference image $\boldsymbol{I}$ and a stack of initial image estimates $\boldsymbol{Q}_1, \dots, \boldsymbol{Q}_M$. Iterative minimisation updates pixels repeatedly, for each selecting the estimate that minimises the perceptual error (6.4). Error diffusion quantises each pixel to the closest estimate, distributing the error to its neighbours based on a kernel $\boldsymbol{\kappa}$. Dithering quantises each pixel in $\boldsymbol{I}$ based on thresholds looked up in a dither mask $\boldsymbol{B}$ (optimised w.r.t. the kernel $\boldsymbol{g}$).

```
function IterativeMinimisation($\boldsymbol{g}$, $\boldsymbol{I}$, $\boldsymbol{Q}_1$, ..., $\boldsymbol{Q}_M$, $\boldsymbol{O}$, $T$)
  $\boldsymbol{O} = \{Q_{1,\text{rand}}, \dots, Q_{N,\text{rand}}\}$        ← Init each pixel to random estimate
  for $T$ iterations do
    for pixel $i = 1..N$ do                                      ← E.g., random or serpentine order
      for estimate $Q_{i,j} \in \{Q_{i,1}, \dots, Q_{i,M}\}$ do
        if $O_i == Q_{i,j}$ reduces $\|\boldsymbol{g} * (\boldsymbol{O} - \boldsymbol{I})\|_2^2$ then
          $O_i = Q_{i,j}$                                        ← Update estimate
function ErrorDiffusion($\boldsymbol{\kappa}$, $\boldsymbol{I}$, $\boldsymbol{Q}_1$, ..., $\boldsymbol{Q}_M$, $\boldsymbol{O}$)
  $\boldsymbol{O} = \boldsymbol{I}$                                    ← Initialise solution to reference
  for pixel $i = 1..N$ do                                        ← E.g., serpentine order
    $O_i^{\text{old}} = O_i$
    $O_i \in \arg\min_{Q_{i,j}} \|O_i^{\text{old}} - Q_{i,j}\|_2^2$
    $\epsilon_i = O_i^{\text{old}} - O_i$                  ↙ Diffuse error $\epsilon_i$ to yet-unprocessed neighbours
    for unprocessed pixel $k$ within support of $\boldsymbol{\kappa}$ around $i$ do
      $O_k \mathrel{+}= \epsilon_i \cdot \kappa_{k-i}$
function Dithering($\boldsymbol{B}$, $\boldsymbol{I}$, $\boldsymbol{Q}_1$, ..., $\boldsymbol{Q}_M$, $\boldsymbol{O}$)
  for pixel $i = 1..N$ do                     ↙ Find tightest interval $[Q_i^{\text{low}}, Q_i^{\text{high}}]$
    $Q_i^{\text{lower}} = \arg\max_{Q_{i,j} : |Q_{i,j}| \le |I_i|} |Q_{i,j}|$                  containing $I_i$
    $Q_i^{\text{upper}} = \arg\min_{Q_{i,j} : |Q_{i,j}| > |I_i|} |Q_{i,j}|$
    if $|I_i| - |Q_i^{\text{lower}}| < B_i \cdot \left(|Q_i^{\text{upper}}| - |Q_i^{\text{low}}|\right)$ then
      $O_i = Q_i^{\text{lower}}$     ↖ Set $O_i$ to $Q_i^{\text{lower}}$ or $Q_i^{\text{upper}}$ using threshold $B_i$
    else
      $O_i = Q_i^{\text{upper}}$
```

6.4.2 Horizontal Search Space

We now describe the second, "horizontal" discrete variant of our minimisation formulation (6.5). It considers a single sample set S_i assigned to each of the N pixels, all represented together as a sample-set image $\boldsymbol{S}$. The search space comprises all possible permutations $\Pi(\boldsymbol{S})$ of these assignments:

$$\boldsymbol{\Omega} = \Pi(\boldsymbol{S}), \text{ with } \boldsymbol{S} = \{S_1, \dots, S_N\}. \tag{6.8}$$

The goal is to find a permutation $\pi(\boldsymbol{S})$ that minimises the perceptual error (6.4). The optimisation problem (6.5) thus takes the form

$$\min_{\pi \in \Pi(\boldsymbol{S})} \|\boldsymbol{g} * (\boldsymbol{Q}(\pi(\boldsymbol{S})) - \boldsymbol{I})\|_2^2. \tag{6.9}$$

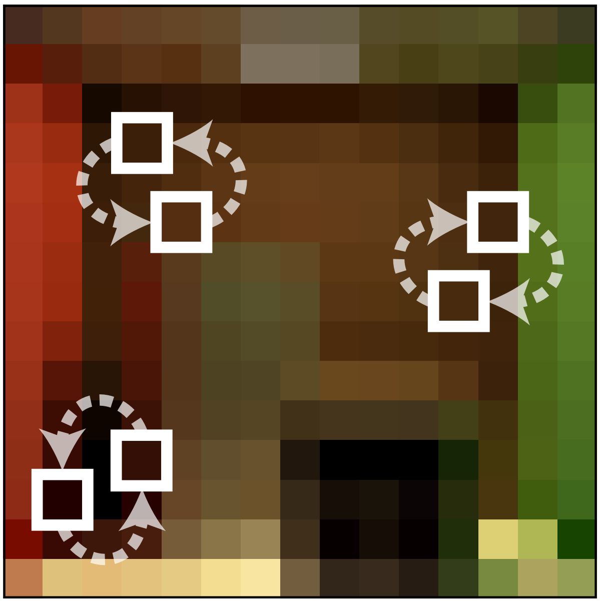


FIGURE 6.4: Horizontal search space optimisation.

We can explore the permutation space $\Pi(\boldsymbol{S})$ by swapping the sample-set assignments between pixels. The minimisation requires updating the image estimate $\boldsymbol{Q}(\pi(\boldsymbol{S}))$ for each permutation $\pi(\boldsymbol{S})$, i.e., after every swap. Such updates are costly as they involve re-sampling both pixels in each of potentially millions of swaps. We need to eliminate these extra rendering invocations during the optimisation to make it practical. To that end, we observe that for pixels solving similar light-transport integrals, swapping their sample sets gives a similar result to swapping their estimates. We therefore restrict the search space to permutations that can be generated through swaps between such (similar) pixels. This enables an efficient optimisation scheme that directly swaps the pixel estimates of an initial rendering $\boldsymbol{Q}(\boldsymbol{S})$.

Error decomposition. Formally, we express the estimate produced by a sample-set permutation in terms of permuting the pixels of the initial rendering: $\boldsymbol{Q}(\pi(\boldsymbol{S})) = \pi(\boldsymbol{Q}(\boldsymbol{S})) + \boldsymbol{\Delta}(\pi)$. The error $\boldsymbol{\Delta}$ is zero when the swapped pixels solve the same integral. Substituting into Eq. (6.9), we can approximate the perceptual error by

$$E(\pi) = \|\boldsymbol{g} * (\pi(\boldsymbol{Q}(\boldsymbol{S})) - \boldsymbol{I} + \boldsymbol{\Delta}(\pi))\|_2^2 \tag{6.10a}$$

$$\approx \|\boldsymbol{g} * (\pi(\boldsymbol{Q}(\boldsymbol{S})) - \boldsymbol{I})\|_2^2 + \|\boldsymbol{g}\|_1^2 \sum_i d(i, \pi(i)) = E_d(\pi), \tag{6.10b}$$

where we write the error $E(\pi)$ as a function of π only, to emphasise that everything else is fixed during the optimisation. In the approximation E_d, the term $d(i, \pi(i))$ measures the dissimilarity between pixel i and the pixel $\pi(i)$ it is relocated to by the permutation. The purpose of this metric is to predict how different we expect the result of re-estimating the pixels after swapping their sample sets to be compared to simply swapping their initial estimates. It can be constructed based on knowledge or assumptions about the image.

Algorithm 2: Given a convolution kernel $\boldsymbol{g}$, a reference image $\boldsymbol{I}$, an initial sample-set assignment $\boldsymbol{S}$, and an image estimate $\boldsymbol{Q}(\boldsymbol{S})$ computed with that assignment, our greedy algorithm iteratively swaps sample-set assignments between neighbouring pixels to minimise the perceptual error E_d (6.10b), producing a permutation π of the initial assignment.

```
function IterativeMinimisation($\boldsymbol{g}$, $\boldsymbol{I}$, $\boldsymbol{S}$, $\boldsymbol{Q}(\boldsymbol{S})$, $T$, $R$, $\pi$)
  $\pi$ = identity permutation                          ← Initialise solution permutation
  for $T$ iterations do
    for pixel $i = 1..N$ do                             ← E.g., random or serpentine order
      $\pi' = \pi$                                      ↙ Find best pixel in neighbourhood to swap with
      for pixel $j$ in $(2R+1)^2$ neighbourhood around $i$ do
        if $E_d(\pi_{i \leftrightarrows j}(\boldsymbol{S})) < E_d(\pi'(\boldsymbol{S}))$ then    ← Eq. (6.10b)
          $\pi' = \pi_{i \leftrightarrows j}$           ← Accept swap as current best
      $\pi = \pi'$
```

Local similarity assumption. Our implementation uses a simple binary dissimilarity function that returns zero when i and $\pi(i)$ are within some distance r and infinity otherwise. We set $r \in [1,3]$; it should ideally be locally adapted to the image smoothness. This allows us to restrict the search space $\Pi(\boldsymbol{S})$ only to permutations that swap adjacent pixels where it is more likely that $\boldsymbol{\Delta}$ is small. More elaborate heuristics could better account for pixel (dis)similarity.

Iterative minimisation. We devise a greedy iterative minimisation scheme for this horizontal formulation, similar to the one in Alg. 1. Given an initial image estimate $\boldsymbol{Q}(\boldsymbol{S})$, produced by randomly assigning a sample set to every pixel, our algorithm goes over all pixels and for each considers swaps within a $(2R+1)^2$ neighbourhood; we use $R = 1$. The swap that brings the largest reduction in the perceptual error E_d is accepted. Algorithm 2 provides pseudocode. In our experiments we run $T = 10$ full-image iterations. As before, the algorithm could be terminated based on the swap reduction rate or the error reduction rate.

The parameter R balances between the cost of one iteration and the amount of exploration it can do. Note that this parameter is different from the maximal relocation distance r in the dissimilarity metric, with $R \leq r$.

Due to the pixel (dis)similarity assumptions, the optimisation can produce some mispredictions, i.e., it may swap the estimates of pixels for which swapping the sample sets produces a significantly different result. Thus the image $\pi(\boldsymbol{Q}(\boldsymbol{S}))$ cannot be used directly as a final estimate. We therefore re-render the image using the optimised permutation π to obtain the final estimate $\boldsymbol{Q}(\pi(\boldsymbol{S}))$.

6.4.3 Discussion

Search space. We discretise the search space $\boldsymbol{\Omega}$ to make the optimisation problem (6.5) tractable. To make it truly practical, it is also necessary to avoid repeated image estimation (i.e., $\boldsymbol{Q}(\boldsymbol{S})$ evaluation) during the search for the solution $\boldsymbol{S}$. Our vertical (6.7) and horizontal (6.9) optimisation variants are formulated specifically with this goal in mind. All methods in Algs. 1 and 2 operate on pre-generated image estimates that constitute the solution search space.

Our vertical formulation takes a collection of M input estimates $\{Q_{i,j} = Q_i(S_{i,j})\}_{j=1}^{M}$ for every pixel i, one for each sample set $S_{i,j}$. Noting that $Q_{i,j}$ are MC estimates of the true pixel value, this collection can be cheaply expanded to a size as large as $2^M - 1$ by taking the average of the estimates in each of its subsets (excluding the empty subset). In practice only a fraction of these subsets can be used, since the size of the power set grows exponentially with M. It may seem that this approach ends up wastefully throwing away most input estimates. But note that these estimates actively participate in the optimisation and provide the space of possible solutions. Carefully selecting a subset per pixel can yield a higher-fidelity result than blindly averaging all available estimates, as we will show repeatedly in Section 6.7.

In contrast, our horizontal formulation builds a search space given just a single input estimate Q_i per pixel. We consciously restrict the space to permutations between nearby pixels, so as to leverage local pixel similarity and avoid repeated pixel evaluation during optimisation. The disadvantage of this approach is that it requires re-rendering the image after optimisation, with uncertain results (due to mispredictions) that can lead to local degradation of image quality. Mispredictions can be reduced by exploiting knowledge about the rendering function $\boldsymbol{Q}(\boldsymbol{S})$, e.g., through depth, normal, or texture buffers. This is further explored in the supplemental of our paper [59] upon which this chapter is based. Additionally, while methods like iterative minimisation (Alg. 2) and dithering (Section 6.5.2) can be adapted to this search space, reformulating other halftoning algorithms such as error diffusion is non-trivial.

A hybrid formulation is also conceivable, taking a single input estimate per pixel (like horizontal methods) and considering a separate (vertical) search space for each pixel constructed by borrowing estimates from neighbouring pixels. Such an approach could benefit from advanced halftoning optimisation methods, but could also suffer from mispredictions and require re-rendering. We leave the exploration of this approach to future work.

Finally, it is worth noting that discretisation is not the only route to practicality. Equation (6.5) can be optimised on the continuous space $\boldsymbol{\Omega}$ if some cheap-to-evaluate proxy for the rendering function is available. Such a continuous approximation may be analytical (based on prior knowledge or assumptions) or obtained by reconstructing a point-wise evaluation. However, continuous-space optimisation can be difficult in high dimensions (e.g., number of light bounces) where non-linearities and non-convexity are exacerbated.

Optimisation strategy. Another important choice is the optimisation method. For the vertical formulation, iterative minimisation provides the best flexibility and quality but is the most computationally expensive. Error diffusion and dithering are faster but only approximately solve Eq. (6.7).

One difference between classical halftoning and our vertical setting is that quantisation levels are non-uniformly distributed and differ between pixels. This further increases the gap in quality between the image-adaptive iterative minimisation and error diffusion (which can correct for these differences) and the non-adaptive dithering, compared to the halftoning setting. The main advantage of dithering is that it involves the kernel $\boldsymbol{g}$ explicitly, while the error-diffusion kernel $\boldsymbol{\kappa}$ cannot be related directly to $\boldsymbol{g}$.

6.5 Practical Application

We now turn to the practical use of our error optimisation framework. In both our discrete formulations from Section 6.4, the search space is determined by a given collection of sample sets $S_{i,j}$ for every pixel i, with $j = 1...M$ (in the horizontal setting $M = 1$). The optimisation is then driven by the corresponding estimates $Q_{i,j}$. We consider two ways to obtain these estimates, leading to different practical trade-offs: (1) direct evaluation of the samples by rendering a given scene and (2) using a proxy for the rendering function. We show how prior works correspond to using either approach within our framework, which helps expose their implicit assumptions.

6.5.1 Surrogate for Ground Truth

The goal of our optimisation is to perceptually match an image estimate to the ground truth $\boldsymbol{I}$ as closely as possible. Unfortunately, the ground truth is unknown in our setting, unlike in halftoning. The best we can do is substitute it with a *surrogate* image $\boldsymbol{I'}$. Such an image can be obtained either from available pixel estimates or by making assumptions about the ground truth. We will discuss specific approaches in the following, but it is already worth noting that all existing error-distribution methods rely on such a surrogate, whether explicitly or implicitly. And since the surrogate guides the optimisation, its fidelity directly impacts the fidelity of the output.

6.5.2 A-posteriori Optimisation

Given a scene and a viewpoint, initial pixel estimates can be obtained by invoking the renderer with the input samples: $Q_{i,j} = Q_i(S_{i,j})$. A surrogate can then be constructed from those estimates; in our implementation we denoise the estimate-average image (Section 6.7.1). Having the estimates and the surrogate, we can run any of the methods in Algs. 1 and 2. Vertical algorithms directly output an image $\boldsymbol{O}$; horizontal optimisation yields a sample-set image $\boldsymbol{S}$ that requires an additional rendering invocation: $\boldsymbol{O} = \boldsymbol{Q}(\boldsymbol{S})$.

This general approach of utilising sampled image information was coined *a-posteriori* optimisation by Heitz and Belcour [103]; they proposed two such methods. Their histogram method operates in a vertical setting, choosing one of the (sorted) estimates for each pixel based on the respective value in a given blue-noise dither mask. Such sampling corresponds to using an implicit surrogate that is the median estimate for every pixel, which is what the mean of the dither mask maps to. Importantly, any one of the estimates for a pixel can be selected, whereas in classical dithering the choice is between the two quantisation levels that tightly envelop the reference value (Section 6.4.1) [218]. Such selection can yield large error, especially for pixels whose corresponding mask values deviate strongly from the mask mean. This produces image fireflies that do not appear if simple estimate averages are taken instead (see Fig. 6.7).

The permutation method of Heitz and Belcour [103] operates in a horizontal setting. Given an image estimate, it finds pixel permutations within small tiles that minimise the distance between the estimates and the values of a target blue-noise mask. This matching transfers the mask's distribution to the image signal rather than to its error. The two are equivalent only when the signal within each tile is constant. The implicit surrogate in this method is thus a tile-wise constant image. In our framework the use of a surrogate is explicit, which enables full control over the quality of the error distribution.

6.5.3 A-priori Optimisation

Optimising perceptual error is possible even in the absence of information about a specific image. In our framework, the goal of such an *a-priori* approach (as coined by Heitz and Belcour [103]) is to compute a sample-set image $\boldsymbol{S}$ by using surrogates for both the ground-truth image $\boldsymbol{I}$ and the rendering function $\boldsymbol{Q}(\boldsymbol{S})$, constructed based on smoothness assumptions. The samples $\boldsymbol{S}$ can then produce a high-fidelity estimate of any image that meets those assumptions.

Lacking prior knowledge, one could postulate that every pixel i has the same rendering function: $Q_i(\cdot) = Q(\cdot)$; the image surrogate $\boldsymbol{I'}$ is thus constant. While in practice this assumption (approximately) holds only locally, the optimisation kernel $\boldsymbol{g}$ is also expected to have compact support. The shape of Q can be assumed to be (piecewise) smooth and approximable by a cheap analytical function Q'.

With the above surrogates in place, we can run our algorithms to optimise a sample-set image $\boldsymbol{S}$. The constant-image assumption makes horizontal algorithms well-suited for this setting as it makes the swapping-error term $\boldsymbol{\Delta}$ in Eq. (6.10a) vanish, simplifying the perceptual error to $E(\pi(\boldsymbol{S})) = \|\boldsymbol{g} * \pi(\boldsymbol{\epsilon}(\boldsymbol{S}))\|_2^2$. This enables the optimisation to consider swaps between *any* two pixels in the error image $\boldsymbol{\epsilon}(\boldsymbol{S})$. That image can be quickly rendered in advance, by invoking the render-function surrogate Q' with the input sample-set image.

Georgiev and Fajardo [93] take a similar approach, with swapping based on simulated annealing. Their empirically motivated optimisation energy uses an explicit (Gaussian) kernel, but instead of computing an error image through a rendering surrogate, it postulates that the distance between two sample sets is representative of the difference between their corresponding pixel errors. Such a smoothness assumption holds for bi-Lipschitz-continuous functions. Their energy can thus be understood to compactly encode properties of a class of rendering functions.

Heitz et al. [104] adopt the approach of Georgiev and Fajardo [93], but their energy function replaces the distance between sample sets by the difference between their corresponding pixel errors. The errors are computed using an explicit render-function surrogate. They optimise for a large number of simple surrogates simultaneously, and leverage a compact representation of Sobol sequences to also support progressive sampling. These two prior works have been related to ours more formally in the supplemental material of our paper [59], on which this chapter is based. The latter also shows how our perceptual error formulation can be incorporated into the method of Heitz et al. [104].

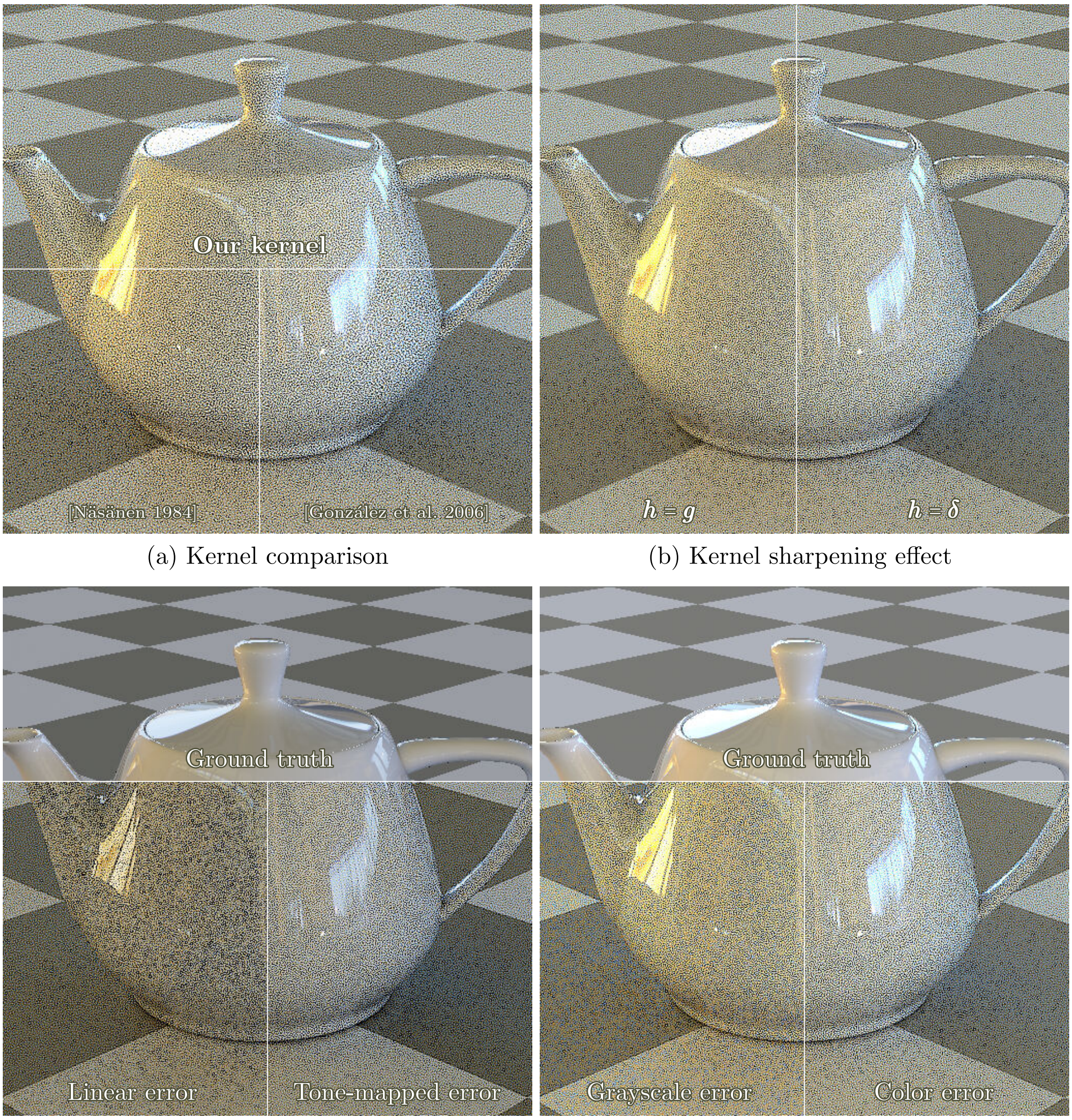
The approach of Ahmed and Wonka [5] performs on-the-fly scrambling of a Sobol sequence applied to the entire image. Image pixels are visited in Morton Z-order modified to breaks its regularity. The resulting sampler diffuses Monte Carlo error over hierarchically nested blocks of pixels giving a perceptually pleasing error distribution. However, the algorithmic nature of this approach introduces more implicit assumptions than prior works, making it difficult to analyse.

Our theoretical formulation and optimisation methods enable the construction of a-priori sampling masks in a principled way. For horizontal optimisation, we recommend using our iterative algorithm (Alg. 2) which can bring significant performance improvement over simulated annealing; such speed-up was reported by Analoui and Allebach [11] for dither-mask construction. Vertical optimisation is an interesting alternative, where for each pixel one of several sample sets would be chosen; this would allow for varying the sample count per pixel. Note that the ranking-key optimisation for progressive sampling of Heitz et al. [104] is a form of vertical optimisation.

6.5.4 Discussion

Our formulation expresses a-priori and a-posteriori optimisation under a common framework that unifies existing methods. These two approaches come with different trade-offs. A-posteriori optimisation utilises sampled image information, and in a vertical setting requires no assumptions except for what is necessary for surrogate construction. It thus has potential to achieve high output fidelity, especially on scenes with complex lighting as it is oblivious to the shape and dimensionality of the rendering function, as first demonstrated by Heitz and Belcour [103]. However, it requires pre-sampling (also post-sampling in the horizontal setting), and the optimisation is sensitive to the surrogate quality.

Making aggressive assumptions allows a-priori optimisation to be performed offline once and the produced samples $\boldsymbol{S}$ to be subsequently used to render any image. This approach resembles classical sample stratification where the goal is also to optimise sample distributions w.r.t. some smoothness expectations. In fact, our a-priori formulation subsumes the per-pixel stratification problem, since the perceptual error is minimised when the error image $\boldsymbol{\epsilon}(\boldsymbol{S})$ has both low magnitude and visually pleasing distribution. Two main factors limit the potential of a-priori optimisation, especially on scenes with non-uniform multi-bounce lighting. One is the general difficulty of optimising sample distributions in high-dimensional spaces. The other is that in such scenes the complex shape of the rendering function, both in screen and sample space, can easily break smoothness assumptions and cause high (perceptual) error.



(a) Kernel comparison

(b) Kernel sharpening effect

(c) Tone mapping (ACES)

(d) Color handling

FIGURE 6.5: (a) Our binomial Gaussian approximation $\boldsymbol{g}$ (3×3 pixels, $\sigma = \sqrt{2/\pi}$) performs on par with state-of-the-art halftoning kernels. (b) Setting the reference-image kernel $\boldsymbol{h}$ in Eq. (6.12) to a zero-width $\boldsymbol{\delta}$ kernel sharpens the output. (c) Incorporating tone mapping via Eq. (6.13). (d) Incorporating colour via Eq. (6.14).

To test the capabilities of our formulation, in the following we focus on the a-posteriori approach. In the supplemental document of our paper [59] we had also explored a-priori optimisation based on our framework. The two approaches can also be combined, e.g., by seeding a-posteriori optimisation with a-priori-optimised samples whose good initial guess can improve the quality and convergence speed.

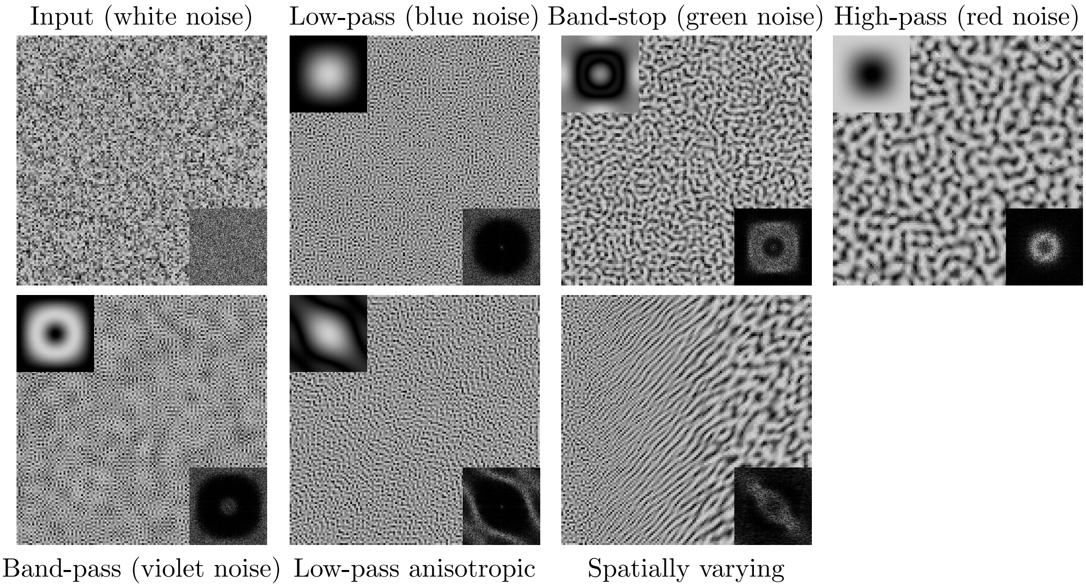


FIGURE 6.6: Our formulation (6.5) allows optimising the error distribution of an image w.r.t. arbitrary kernels. Here we adapt our horizontal iterative minimisation (Alg. 2) to directly swap the pixels of a white-noise input image. Insets show the power spectra of the target kernels (top left) and the optimised images (bottom right).

6.6 Extensions

Our perceptual error formulation (6.4) approximates the effect of the HVS PSF through kernel convolution. In this section we analyse the relationship between kernel and viewing distance, as well as the impact of the kernel shape on the error distribution. We also present extensions that account for the HVS non-linearities in handling tone and colour.

Kernels and viewing distance As discussed in Section 6.3.1, the PSF is usually modelled over a range of viewing distances. The effect of the PSF depends on the frequencies of the viewed signal and the distance from which it is viewed. Pappas and Neuhoff [183] have found that the Gaussian is a good approximation to the PSF in the context of halftoning. They derived its standard deviation σ in terms of the minimum viewing distance for a given screen resolution:

$$\sigma = \frac{0.00954}{\tau}, \quad \text{where} \quad \tau = \frac{180}{\pi} 2 \arctan\left(\frac{1}{2RD}\right). \tag{6.11}$$

Here τ is the visual angle between the centres of two neighbouring pixels (in degrees) for screen resolution R (in 1/inches) and viewing distance D (in inches). The minimum viewing distance for a given standard deviation and resolution can be contained via the inverse formula: $D = \left(2R\tan\left(\frac{\pi}{180}\frac{0.00954}{2\sigma}\right)\right)^{-1}$. Larger σ values correspond to larger observer distances; we demonstrate the effect of that in Fig. 6.2 where the images become increasingly blurrier. In Fig. 6.5a, we compare that Gaussian kernel to two well-established PSF models from the halftoning literature [94, 174]. We have found the differences between all three to be negligible; we use the cheaper to evaluate Gaussian in all our experiments.

Decoupling the viewing distances. Being based on the perceptual models of the HVS [11, 223], our formulation (6.4) assumes that the estimate $\boldsymbol{Q}$ and the reference $\boldsymbol{I}$ are viewed from the same (range of) distance(s). The two distances can be decoupled by applying different kernels to $\boldsymbol{Q}$ and $\boldsymbol{I}$:

$$E = \|\boldsymbol{g} * \boldsymbol{Q} - \boldsymbol{h} * \boldsymbol{I}\|_2^2. \tag{6.12}$$

Minimising this error makes $\boldsymbol{Q}$ appear from some distance $D_{\boldsymbol{g}}$ similar to $\boldsymbol{I}$ seen from a different distance $D_{\boldsymbol{h}}$. The special case of using a Kronecker delta kernel $\boldsymbol{h} = \boldsymbol{\delta}$, i.e., with the reference $\boldsymbol{I}$ seen from up close, yields $E = \|\boldsymbol{g} * \boldsymbol{Q} - \boldsymbol{I}\|_2^2$. This has been shown to have an edge enhancing effect [12, 183] which we show in Fig. 6.5b. We use $\boldsymbol{h} = \boldsymbol{\delta}$ in all our experiments.

Tonemapping. Considering that the optimised image will be viewed on media with limited dynamic range (e.g., screen or paper), we can incorporate a tone-mapping operator $\mathcal{T}$ into the perceptual error (6.4):

$$E = \|\boldsymbol{g} * \boldsymbol{\epsilon}_{\mathcal{T}}\|_2^2 = \|\boldsymbol{g} * (\mathcal{T}(\boldsymbol{Q}) - \mathcal{T}(\boldsymbol{I}))\|_2^2. \tag{6.13}$$

Doing this also bounds the per-pixel error $\boldsymbol{\epsilon}_{\mathcal{T}} = \mathcal{T}(\boldsymbol{Q}) - \mathcal{T}(\boldsymbol{I})$, suppressing outliers and making the optimisation more robust in scenes with high dynamic range. We illustrate this improvement in Fig. 6.5c, where an ACES [15] tone-mapping operator is applied to the optimised image. Optimising w.r.t. the original perceptual error (6.4) yields a noisy and overly dark image compared to the tone-mapped ground truth. Accounting for tone mapping in the optimisation through Eq. (6.13) yields a more faithful result.

Colourhandling. While the HVS reacts more strongly to luminance than colour, ignoring chromaticity entirely (e.g., by computing the error image $\boldsymbol{\epsilon}$ from per-pixel luminances) can have a negative effect on the distribution of colour noise in the image. To that end, we can penalise the perceptual error of each colour channel $c \in C$ separately:

$$E = \sum_{c \in C} \lambda_c \|\boldsymbol{g}_c * (\boldsymbol{Q}_c - \boldsymbol{I}_c)\|_2^2, \tag{6.14}$$

where λ_c is a per-channel weight. In our experiments, we use an RGB space $C = \{\mathrm{r}, \mathrm{g}, \mathrm{b}\}$, set $\lambda_c = 1$, and use the same kernel $\boldsymbol{g}_c = \boldsymbol{g}$ for every channel. Figure 6.5d shows the improvement in colour noise over using greyscale perceptual error. Depending on the colour space, the per-channel kernels may differ (e.g., YCbCr) [223].

As an alternative, one could decouple the channels from the input estimates and optimise each channel separately, assembling the results into a colour image. In a vertical setting, this decoupling extends the optimisation search space size from M to $M^{|C|}$.

Kernel shape impact. To test the robustness of our framework, we analyse kernels with spectral characteristics other than isotropic blue-noise in Fig. 6.6. We run our iterative pixel-swapping algorithm (Alg. 2) to optimise the shape of a white-noise input, which produces a spectral distribution inverse to that of the target kernel. The rightmost image in the figure shows the result of using a spatially varying kernel that is a convex combination between a low-pass Gaussian and a high-pass anisotropic kernel, with the interpolation parameter varying horizontally across the image. Our algorithm can adapt the noise shape well.

6.7 Experiments

We now present empirical validation of our error optimisation framework in the a-posteriori setting described in Section 6.5.2. We render static images and animations of several scenes, comparing our algorithms to those of Heitz and Belcour [103].

6.7.1 Setup

Perceptual error model. We build a perceptual model by combining all extensions from Section 6.6. Our estimate-image kernel $\boldsymbol{g}$ is a binomial approximation of a Gaussian [151]. For performance reasons and to allow smaller viewing distances we use a 3×3-pixel kernel with standard deviation $\sigma = \sqrt{2/\pi}$ (see Fig. 6.5a). Plugging this σ value into the inverse of Eq. (6.11), the corresponding minimum viewing distance is $D = 4792/R$ inches for a screen resolution of R dpi (e.g., 16 inches at 300 dpi). We recommend viewing from a larger distance, to reduce the effect of our 3×3 kernel discretisation. We use a Dirac reference-image kernel: $\boldsymbol{h} = \boldsymbol{\delta}$, and incorporate a simple tone-mapping operator $\mathcal{T}$ that clamps pixel values to $[0, 1]$. The final error model reads:

$$E = \sum_{c \in \{\mathrm{r,g,b}\}} \| \boldsymbol{g} * \mathcal{T}(\boldsymbol{Q}_c) - \boldsymbol{\delta} * \mathcal{T}(\boldsymbol{I}'_c) \|_2^2, \tag{6.15}$$

where $\boldsymbol{I}'$ is the surrogate image whose construction we describe below. For dithering we convert RGB colours to luminance, which reduces the number of components in the error (6.15) to one.

Methods. We compare our four methods from Algs. 1 and 2 to the histogram and permutation of Heitz and Belcour [103]. For our vertical and horizontal iterative minimisations we set the maximum iteration count to 100 and 10 respectively. For error diffusion we use the kernel of Floyd and Steinberg [85] and for dithering we use a void-and-cluster mask [232]. For our horizontal iterative minimisation we use a search radius $R = 1$ and allow pixels to travel within a disk of radius $r = 1$ from their original location in the dissimilarity metric. For the permutation method of Heitz and Belcour [103] we obtained best results with tile size 8×8. (Our $r = 1$ approximately corresponds to their tile size 3×3.)

Rendering. All scenes were rendered with PBRT [193] using unidirectional or bidirectional path tracing. None of the methods depend on the sampling dimensionality, though we set the maximum path depth to 5 for all scenes to maintain reasonable rendering times. The ground-truth images have been generated using a Sobol sampler with at least 1024 samples per pixel (spp); for all test renders we use a random sampler. To facilitate numerical-error comparisons between the different methods, we trace the primary rays through the pixel centres.

Surrogate construction. To build a surrogate image for our methods, we filter the per-pixel averaged input estimates using Intel Open Image Denoise [116] which also leverages surface-normal and albedo buffers, taking about 0.5 sec for a 512×512 image. Recall that the methods of Heitz and Belcour [103] utilise implicit surrogates.

Image-quality metrics. We evaluate the quality of some of our results using the HDR-VDP-2 perceptual metric [161], with parameters matching our binomial kernel. We compute error-detection probability maps which indicate the likelihood for a human observer to notice a difference from the ground truth.

Additionally, we analyse the local blue-noise quality of the error image $\boldsymbol{\epsilon} = \mathcal{T}(\boldsymbol{Q}) - \mathcal{T}(\boldsymbol{I})$. We split the image into tiles of 32×32 pixels and compute the Fourier power spectrum of each tile. For visualisation purposes, we apply a standard logarithmic transform $c \ln(1 + |\hat{\epsilon}|)$ to every resulting pixel value $\hat{\epsilon}$ and compute the normalisation factor c per tile so that the maximum final RGB value within the tile is $(1, 1, 1)$. Note that the error image $\boldsymbol{\epsilon}$ is computed w.r.t. the ground truth $\boldsymbol{I}$ and not the surrogate, which quantifies the blue-noise distribution objectively. The supplemental material of our paper [59] contains images of the tiled power spectra for all experiments.

We compare images quantitatively via traditional MSE as well as a metric derived from our perceptual error formulation. Our *perceptual MSE* (pMSE) evaluates the error (6.15) of an estimate image w.r.t. the ground truth, normalised by the number of pixels N and channels C: $\text{pMSE} = \frac{E}{N \cdot C}$. It generalises the MSE with a perceptual, i.e., non-delta, kernel $\boldsymbol{g}$. Table 6.1 summarises the results.

6.7.2 Rendering comparisons

All methods. Figure 6.7 shows an equal-sample comparison of all methods. Vertical methods select one of the 4 input samples per pixel; horizontal methods are fed a 2-spp average for every pixel, and another 2 spp are used to render the final image after optimisation. Our methods consistently perform best visually, with the vertical iterative minimisation achieving the lowest perceptual error, as corroborated by the HDR-VDP-2 detection maps. Error diffusion is not far behind in quality and is the fastest of all methods along with dithering. The latter is similar to Heitz and Belcour's histogram method but yields a notably better result thanks to using a superior surrogate and performing the thresholding as in the classical halftoning setting (see Section 6.5.2). Horizontal methods perform worse due to noisier input data (half spp) and worse surrogates derived from it, and also mispredictions (which necessitate re-rendering).

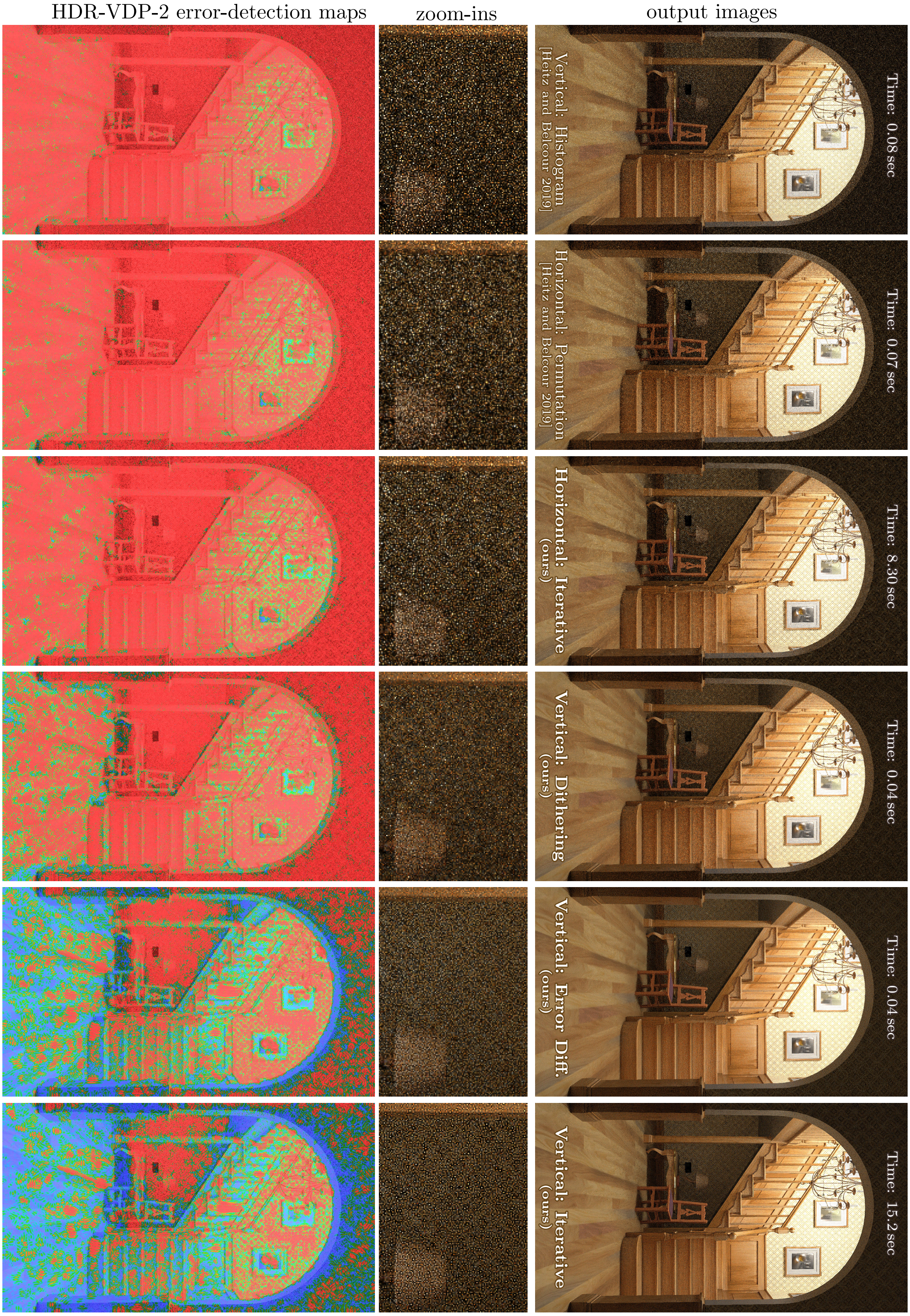


FIGURE 6.7: Comparison of our algorithms against the permutation and histogram methods of Heitz and Belcour [103] with equal total sampling cost of 4 spp. Bottom row shows HDR-VPD-2 error-detection maps (blue is better, i.e., lower detection probability). Our vertical iterative minimisation achieves highest fidelity. Error diffusion produces similar quality. Our horizontal iterative optimisation does better than the permutation method.

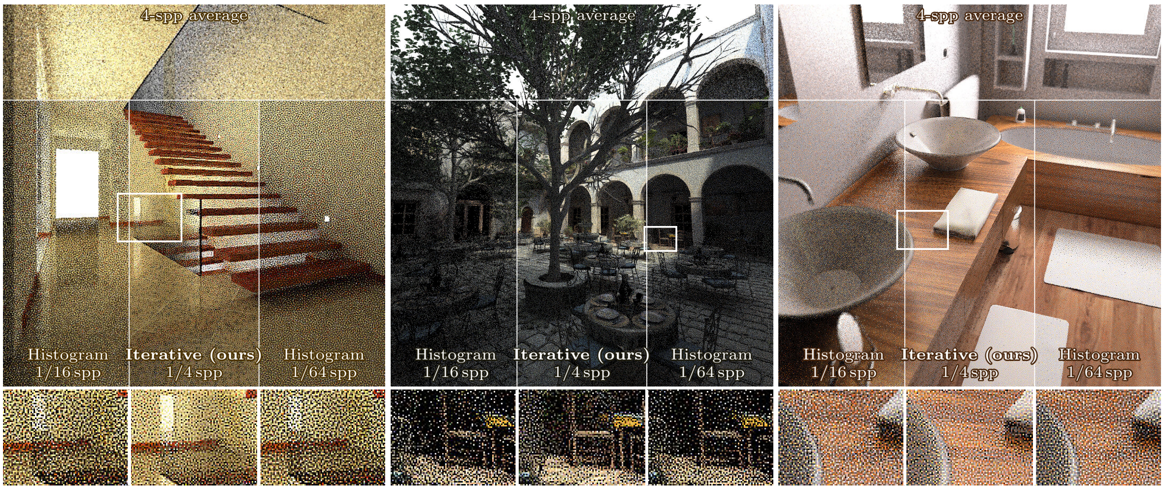


FIGURE 6.8: With a search space of only 4 spp, our vertical iterative minimisation outperforms histogram sampling [103] with 16× more input samples. Please zoom in to fully appreciate the differences; the full-size images are included in the supplemental material of our paper [59].

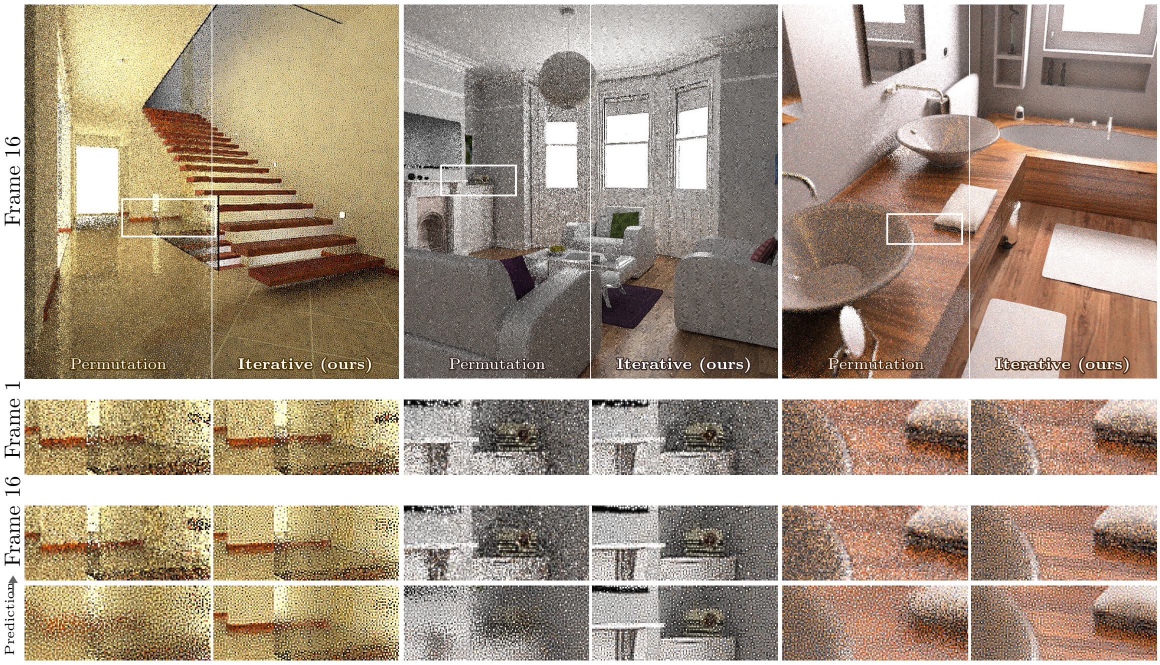


FIGURE 6.9: Comparison of our horizontal iterative minimisation against the permutation method of Heitz and Belcour [103] (with retargeting) on 16-frame sequences of static scenes rendered at 4 spp. Our method does a better job at improving the error distribution frame-to-frame.

Ours still uses a better surrogate than Heitz and Belcour's permutation and is also able to better fit to it.

Vertical methods. In Fig. 6.8 we compare our vertical iterative minimisation to the histogram sampling of Heitz and Belcour [103]. Both select one of several input samples (i.e., estimates) for each pixel. Our method produces a notably better result even when given 16× fewer samples to choose from. The perceptual error of histogram sampling does not vanish with increasing sample count. It dithers pixel *intensity* rather than pixel error, thus more samples help improve the intensity distribution but not the error magnitude.

Figure 6.10 shows our most capable method: vertical iterative minimisation with search space extended to the power set of the input samples (with size $2^4 - 1 = 15$ for 4 input spp; see Section 6.4.3). We compare surrogate-driven optimisation against the best-case result—optimisation w.r.t. the ground truth. Both achieve high fidelity, with little difference between them and with pronounced local blue-noise error distribution corroborated by the tiled power spectra.

Horizontal methods & animation. For rendering static images, horizontal methods are at a disadvantage compared to vertical ones due to the required post-optimisation re-rendering. As Heitz and Belcour [103] note, in an animation setting this sampling overhead can be mitigated by reusing the result of one frame as the initial estimate for the next. In Fig. 6.9 we compare their permutation method to our horizontal iterative minimisation. For theirs we shift a void-and-cluster mask in screen space per frame and apply retargeting, and for ours we traverse the image pixels in different random order. We intentionally keep the scenes static to test the methods' best-case abilities to improve the error distribution over frames.

Starting from a random initial estimate, our method can benefit from a progressively improving surrogate that helps fine-tune the error distribution via localised pixel swaps. The permutation method operates in greyscale within static non-overlapping tiles. This prevents it from making significant progress after the first frame. While mispredictions cause local deviations from blue noise in both results, these are stronger in the permutation method's. This is evident when comparing the corresponding prediction images—the results of optimisation right before re-rendering. The permutation's retargeting pass breaks the blocky image structure caused by tile-based optimisation but increases the number of mispredictions.

The supplemental video to our published work [59] shows animations with all methods, where vertical ones are fed a new random estimate per frame. Even without accumulating information over time, these consistently beat the two horizontal methods. The latter suffer from mispredictions under fast motion and perform similarly to one another, though ours remains superior in the presence of temporal smoothness. Mispredictions could be eliminated by optimising frames independently and splitting the sampling budget into optimisation and re-rendering halves (as in Fig. 6.7), though at the cost of reduced sampling quality.

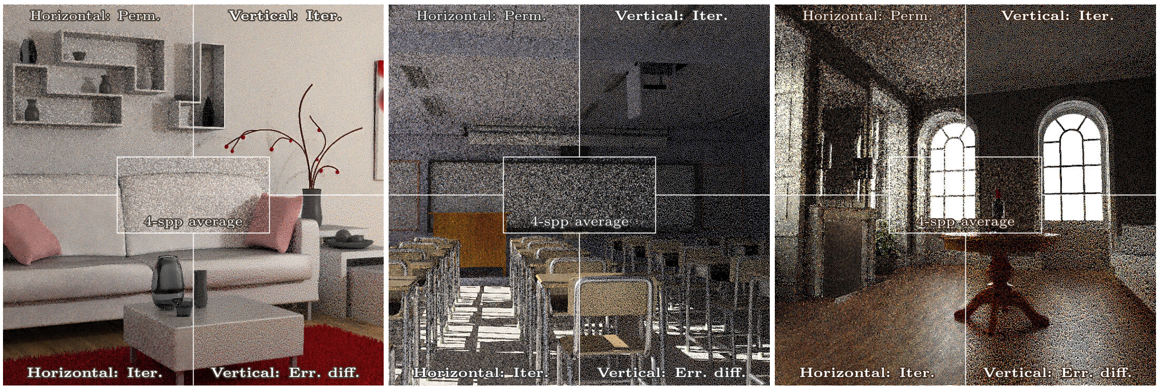


FIGURE 6.10: Comparison of our methods against the permutation approach of Heitz and Belcour [103] at 4 spp; for the horizontal methods we show the result of the 16th frame of static-scene rendering. Our two iterative minimisation algorithms yield the best quality, while error diffusion is fastest (see Tables 6.1 and 6.2).

Additional comparisons. Figure 6.10 shows additional results from our horizontal and vertical minimisation and error diffusion. We compare these to the permutation method of Heitz and Belcour [103] which we found to perform better than their histogram approach on static scenes at equal sampling rates. For the horizontal methods we show the results after 16 iterations. Our methods again yield lower error, subjectively and numerically (see Tables 6.1 and 6.2).

6.8 Discussion

6.8.1 Bias Towards Surrogate

While ultimately we want to optimise w.r.t. the ground-truth image, in practice we have to rely on a surrogate. In our experiments we use reasonably high-quality surrogates, shown in Fig. 6.13, to best demonstrate the capabilities of our framework. But when using a surrogate of low quality, fitting too closely to it can produce an estimate with artifacts. In such cases less aggressive fitting may yield *lower* perceptual error. We augment the perceptual error with a term that penalises deviations from the initial estimate $\boldsymbol{Q}_{\text{init}}$ (which case of vertical optimisation is obtained by averaging the input per-pixel estimates):

$$E_{\mathcal{C}} = (1-\mathcal{C})\|\boldsymbol{g}\|_1^2\|\boldsymbol{Q}-\boldsymbol{Q}_{\text{init}}\|_2^2 + \mathcal{C}\,E. \tag{6.16}$$

The parameter $\mathcal{C} \in [0,1]$ encodes our confidence in the surrogate quality. Setting $\mathcal{C} = 1$ reverts to the original formulation (6.15), while optimising with $\mathcal{C} = 0$ yields the initial image estimate $\boldsymbol{Q}_{\text{init}}$. Optimising w.r.t. this energy can also be interpreted as projecting the surrogate onto the space of Monte Carlo estimates in $\boldsymbol{\Omega}$, with control over the fitting power of the projection via $\mathcal{C}$.

In Fig. 6.11, we plug the extended error formulation (6.16) into our vertical iterative minimisation. The results indicate that the visually best result is achieved for different

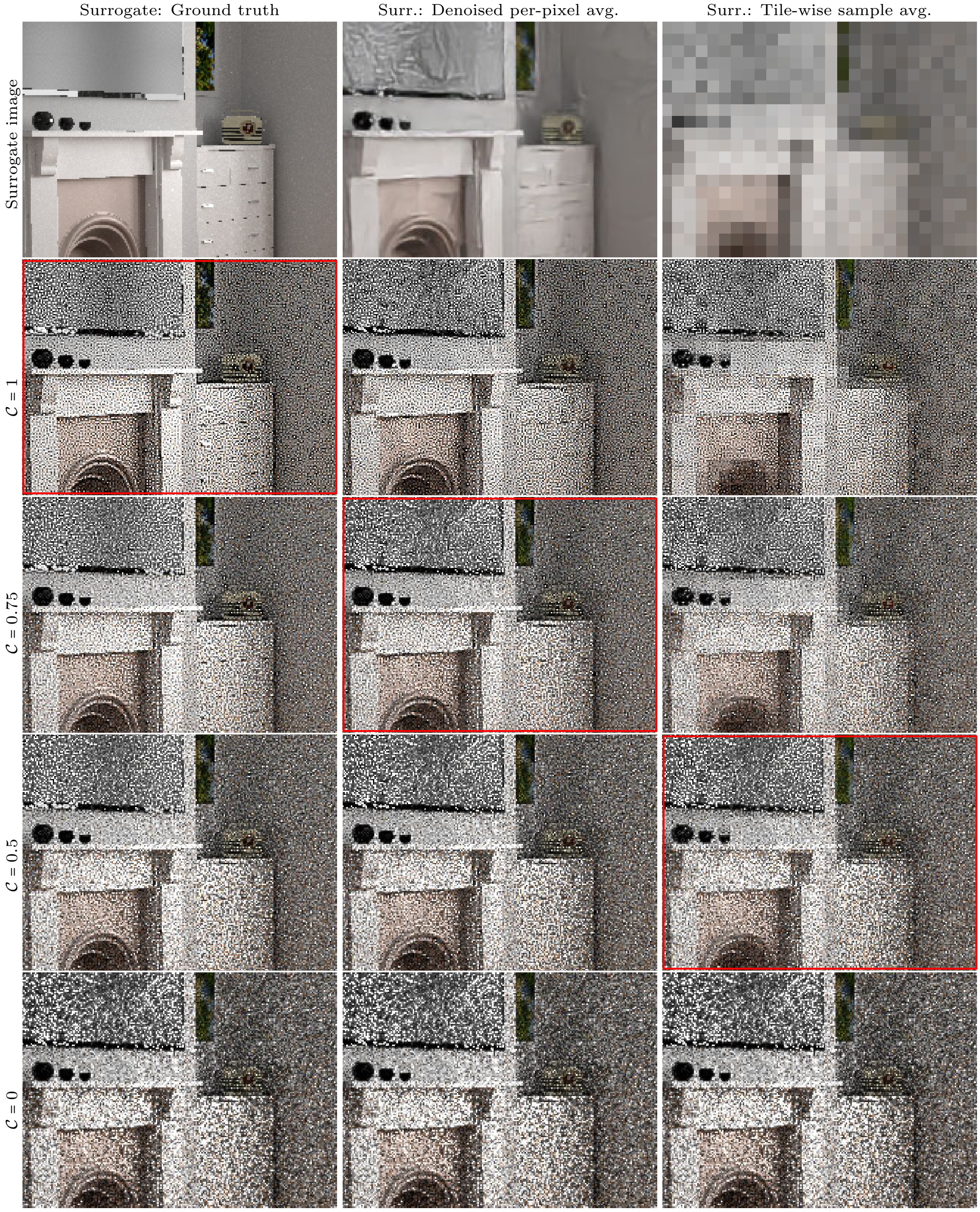


FIGURE 6.11: Balancing our iterative optimisation between the surrogate (top row) and the initial estimate (bottom row) via the parameter $\mathcal{C}$ from Eq. (6.16). For high-quality surrogates (left and middle columns), the best result is achieved for values of $\mathcal{C}$ close to 1. In contrast, strong structural artifacts (right column) call for lowering $\mathcal{C}$ to avoid fitting too closely to the surrogate. The (subjectively) best image in each column is outlined in red.

values of $\mathcal{C}$ depending on the surrogate quality. Specifically, when optimising w.r.t. the ground truth, the fitting should be most aggressive: $\mathcal{C} = 1$. Conversely, if the surrogate contains structural artifacts, the optimisation should be made less biased to it, e.g., by setting $\mathcal{C} = 0.5$. Other ways to control this bias are using a more restricted search space (e.g., non-power-set) and capping the number of minimisation iterations of our methods. Note that the methods of Heitz and Belcour [103] rely on implicit surrogates and energies and thus provide no control over this trade-off. We have found that their permutation method generally avoids tiling artifacts induced by their piecewise constant surrogate due to the retargeting step blurring the prediction image (shown in Fig. 6.9 zoom-ins); however, this blurring adds mispredictions which deteriorate the final image quality. Our methods provide better fits, target the error explicitly, and are much superior when the surrogate is good. With a bad surrogate, ours can be controlled to never do worse than theirs.

6.8.2 Denoising

Our images are optimised for eliminating error and preserving features when blurred with a given kernel. This blurring can be seen as a simple form of denoising, and it is reasonable to expect that the images are also better suited for general-purpose denoising than traditional white-noise renderings are [24, 103]. However, we have found that obtaining such benefit is not straightforward.

In Fig. 6.12 we run Intel Open Image Denoise on the results from our vertical iterative minimisation. On the left scene, the input samples ① have white-noise image distribution with large magnitude; feeding their per-pixel averages to the denoiser, it cannot reliably separate the signal from the noise and produces conspicuous artifacts. Using this denoised image ② as a surrogate for our optimisation yields a "regularised" version ③ of the input that is easier for the denoiser to subsequently filter ④. This process can be seen as projecting the initial denoised image back onto the space of exact per-pixel estimates (while minimising the pMSE) whose subsequent denoising avoids artifacts. Note that obtaining this improved result requires no additional pixel sampling.

On the right scene in Fig. 6.12, the moderate input-noise level is easy for the denoiser to clean while preserving the faint shadow on the wall. Our optimisation subsequently produces an excellent result which yields a high-fidelity image when convolved with the optimisation kernel $\boldsymbol{g}$. Yet that same result is ruined by the denoiser which eradicates the shadow, even though subjectively its signal-to-noise ratio is higher than that of the input image. Overall, the denoiser blurs our result ③ aggressively on both scenes, eliminating not only the high-frequency noise but also lower-frequency signal not present in auxiliary input feature buffers (depth, normals, etc).

It should not be too surprising that an image optimised for one smoothing kernel does not always yield good results when filtered with other kernels. As an example, Fig. 6.6 shows clearly that the optimal noise distribution varies significantly across different kernels. While our kernel $\boldsymbol{g}$ has narrow support and fixed shape, denoising kernels vary wildly over the image and are inferred from the input in order to preserve features. Importantly, modern kernel-inference models (like in the used denoiser) are designed (or

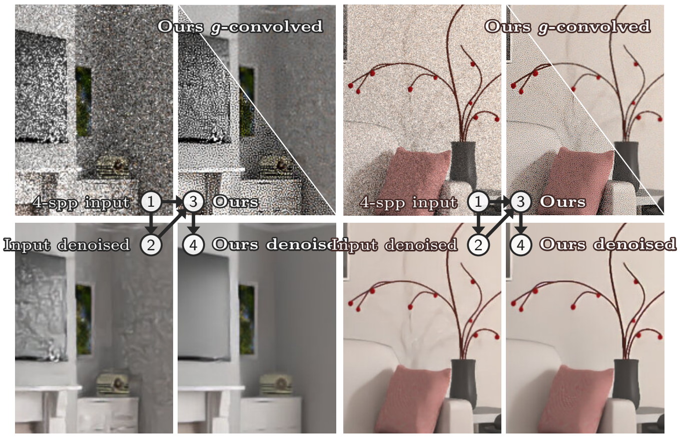


FIGURE 6.12: By regularising a noisy input, our optimisation can help a denoiser avoid producing artifacts (left scene), even though it targets a different (perceptual) smoothing kernel $\boldsymbol{g}$. However, it can also cause elimination of image features during denoising (right scene, the shadow).

trained) to expect mutually uncorrelated pixel estimates [116]. This white-noise-error assumption can also yield wide smoothing kernels that are unnecessarily aggressive for blue-noise distributions; we suspect this is what hinders the denoiser from detecting features present in our optimised results whose pixels are highly correlated.

Our firm belief is that denoising could consistently benefit from error optimisation, though that would require better coordination between the two. One avenue for future work would be to tailor the optimisation to the kernels employed by a target denoiser. Conversely, denoising could be adapted to ingest correlated pixel estimates with high-frequency error distribution; this would enable the use of less aggressive smoothing kernels (see Fig. 6.2) and facilitate feature preservation. As a more immediate treatment, image features could be enhanced before or after our optimisation to mitigate the risk of them being eliminated by denoising.

6.8.3 Performance and Utility

Throughout our experiments, we have found that the tested algorithms rank in the following order in terms of increasing ability to minimise perceptual error on static scenes at equal sampling cost: histogram sampling, our dithering, permutation, our error diffusion, our horizontal iterative, our vertical iterative. The three lowest-ranked methods employ some form of dithering which by design assumes (a) constant image signal and (b) equispaced quantisation levels shared by all pixels. The latter assumption is severely broken in the rendering setting where the “quantisation levels” arise from (random) pixel estimation. Our vertical methods (dithering, error diffusion, iterative) are more practical than the histogram sampling of Heitz and Belcour [103] as they can achieve high fidelity with a much lower input-sample count. Horizontal algorithms are harder to control due to their mispredictions which are further exacerbated when reusing estimates across frames in dynamic scenes.

Method	**Bathroom**		**Classroom**		**Gray Room**		**Living Room**	
	MSE	pMSE	MSE	pMSE	MSE	pMSE	MSE	pMSE
	$\times 10^{-2}$	$\times 10^{-3}$	$\times 10^{-2}$	$\times 10^{-3}$	$\times 10^{-2}$	$\times 10^{-2}$	$\times 10^{-2}$	$\times 10^{-3}$
Random (4-spp average)	1.40	3.15	3.13	7.91	7.91	3.02	3.37	5.61
Vertical: Histogram [103] (1/4 spp)	3.58	6.29	7.11	13.08	11.49	6.67	5.75	9.88
Vertical: Error diffusion (1/4 spp)	1.22	2.27	4.91	7.03	8.76	2.82	2.08	2.31
Vertical: Dithering (1/4 spp)	1.31	3.31	4.36	11.63	8.46	5.07	2.27	4.43
Vertical: Iterative (1/4 spp)	2.32	2.02	6.00	6.10	9.07	2.97	4.32	1.86
Vertical: Iterative (power set, 1/15 "spp")	1.26	1.66	3.12	4.91	7.53	2.82	2.46	1.13
Horizontal: Permut. [103] (frame 16, 4 spp)	1.40	2.79	3.15	7.25	7.90	2.84	3.38	3.14
Horizontal: Iterative (frame 16, 4 spp)	1.52	2.06	3.83	5.31	8.34	2.41	3.59	1.59
Random (16-spp average)	0.49	1.47	1.55	4.89	3.77	1.04	1.23	2.18
Vertical: Histogram [103] (4/16 spp)	1.40	2.37	3.12	6.20	7.88	2.72	3.36	3.57
Vertical: Error diffusion (4/16 spp)	0.41	1.20	0.94	3.85	4.00	0.87	0.86	1.07
Vertical: Dithering (4/16 spp)	0.50	1.52	1.15	4.69	4.12	1.36	1.09	1.82
Vertical: Iterative (4/16 spp)	0.90	1.10	2.03	3.35	5.17	0.84	2.30	0.84

Method	**Modern Hall**		**San Miguel**		**Staircase**		**White Room**	
	MSE	pMSE	MSE	pMSE	MSE	pMSE	MSE	pMSE
	$\times 10^{-2}$	$\times 10^{-2}$	$\times 10^{-2}$	$\times 10^{-3}$	$\times 10^{-3}$	$\times 10^{-3}$	$\times 10^{-2}$	$\times 10^{-3}$
Random (4-spp average)	5.22	1.70	3.58	8.92	8.88	5.60	2.78	7.98
Vertical: Histogram [103] (1/4 spp)	11.43	3.60	6.84	16.52	18.90	6.69	5.75	14.09
Vertical: Error diffusion (1/4 spp)	4.86	1.33	5.07	8.50	6.87	5.08	2.19	5.16
Vertical: Dithering (1/4 spp)	5.25	1.80	3.74	11.19	7.80	5.36	2.51	7.95
Vertical: Iterative (1/4 spp)	7.15	1.29	5.51	7.05	10.50	4.45	3.98	5.00
Vertical: Iterative (power set, 1/15 "spp")	4.55	1.18	3.31	5.85	7.08	4.31	2.26	4.58
Horizontal: Permut. [103] (frame 16, 4 spp)	5.21	1.51	3.59	8.51	8.87	5.40	2.72	6.73
Horizontal: Iterative (frame 16, 4 spp)	5.46	1.18	3.94	7.31	7.67	4.30	2.93	4.72
Random (16-spp average)	2.14	0.80	1.10	4.67	3.39	3.78	1.35	3.62
Vertical: Histogram [103] (4/16 spp)	5.23	1.48	3.52	6.82	7.13	4.09	2.77	5.77
Vertical: Error diffusion (4/16 spp)	1.68	0.66	1.33	4.70	2.76	3.69	0.73	2.13
Vertical: Dithering (4/16 spp)	1.93	0.83	1.49	5.38	3.09	3.73	0.91	2.98
Vertical: Iterative (4/16 spp)	3.03	0.64	2.39	4.02	4.46	3.14	1.75	1.99

Table 6.1: MSE and pMSE (Section 6.7.1) metrics for various methods (ours in bold) and scenes. For horizontal methods we show the metrics for the 16th frame of static-scene rendering. In each section we highlight the lowest error number per column. For the same number of samples per pixel (spp), our methods consistently outperform those of Heitz and Belcour [103]—the current state of the art, except our dithering can do worse than their permutation method.

Method	**Bathroom**	**Classroom**	**Gray Room**	**Living Room**
Vertical: Histogram [103] ($^1/_4$ spp)	0.06	0.07	0.11	0.06
Vertical: Error diffusion ($^1/_4$ spp)	0.04	0.03	0.04	0.04
Vertical: Dithering ($^1/_4$ spp)	0.04	0.03	0.04	0.04
Vertical: Iterative ($^1/_4$ spp)	18.44	111.41	12.82	15.26
Vertical: Iterative (power set, $^1/_{15}$ "spp")	95.09	404.12	59.69	83.41
Horizontal: Permutation [103] (frame 16)	0.10	0.10	0.10	0.11
Horizontal: Iterative (frame 16)	23.04	21.57	22.00	30.08

Method	**Modern Hall**	**San Miguel**	**Staircase**	**White Room**
Vertical: Histogram [103] ($^1/_4$ spp)	0.02	0.09	0.08	0.06
Vertical: Error diffusion ($^1/_4$ spp)	0.01	0.06	0.04	0.04
Vertical: Dithering ($^1/_4$ spp)	0.01	0.05	0.04	0.04
Vertical: Iterative ($^1/_4$ spp)	5.43	29.09	15.21	19.45
Vertical: Iterative (power set, $^1/_{15}$ "spp")	23.93	137.89	35.39	102.05
Horizontal: Permutation [103] (frame 16)	0.03	0.21	0.10	0.14
Horizontal: Iterative (frame 16)	8.48	36.36	23.78	22.76

TABLE 6.2: Optimisation run times (in seconds) for various methods (ours in bold) and scenes using 4 input samples per pixel (spp), excluding sampling and surrogate construction. For horizontal methods we report the average time over 16 frames. Our error diffusion and dithering avoid sorting and are fastest; though dithering-based, Heitz and Belcour's approaches use sorting. Our iterative minimisation methods are slowest (but can be sped up; see Section 6.8.3).

Our iterative minimisations can best adapt to the input and also directly benefit from the extensions in Section 6.6 (unlike all others). However, they are also the slowest, as evident in Table 6.2. Fortunately, they can be sped up by several orders of magnitude through additional optimisations from halftoning literature [11, 135].

Error diffusion is often on par with vertical iterative minimisation in quality and with dithering-based methods in run time. In a single-threaded implementation it can outperform all others; parallel error-diffusion variants exist too [167].

Practical utility. Our methods can enhance the perceptual fidelity of static and dynamic renderings as demonstrated by our experiments. For best results and maximum flexibility, we suggest using our vertical iterative optimisation, optionally with the efficiency improvements mentioned above. Figure 6.11 illustrates that in practical scenarios (middle and right columns) this method can improve upon both the surrogate (top row) and the input-estimate average (bottom row) for a suitable value of the confidence parameter $\mathcal{C}$. For maximum efficiency we recommend using our vertical error diffusion. To obtain a surrogate, we recommend regularising the input estimates via fast denoising or more basic bilateral or non-local-means filtering. Our optimisation can then be interpreted as reducing bias or artifacts in such denoised images (see Fig. 6.11). Simple denoising of the result may yield better quality than traditional aggressive denoising of the input samples.

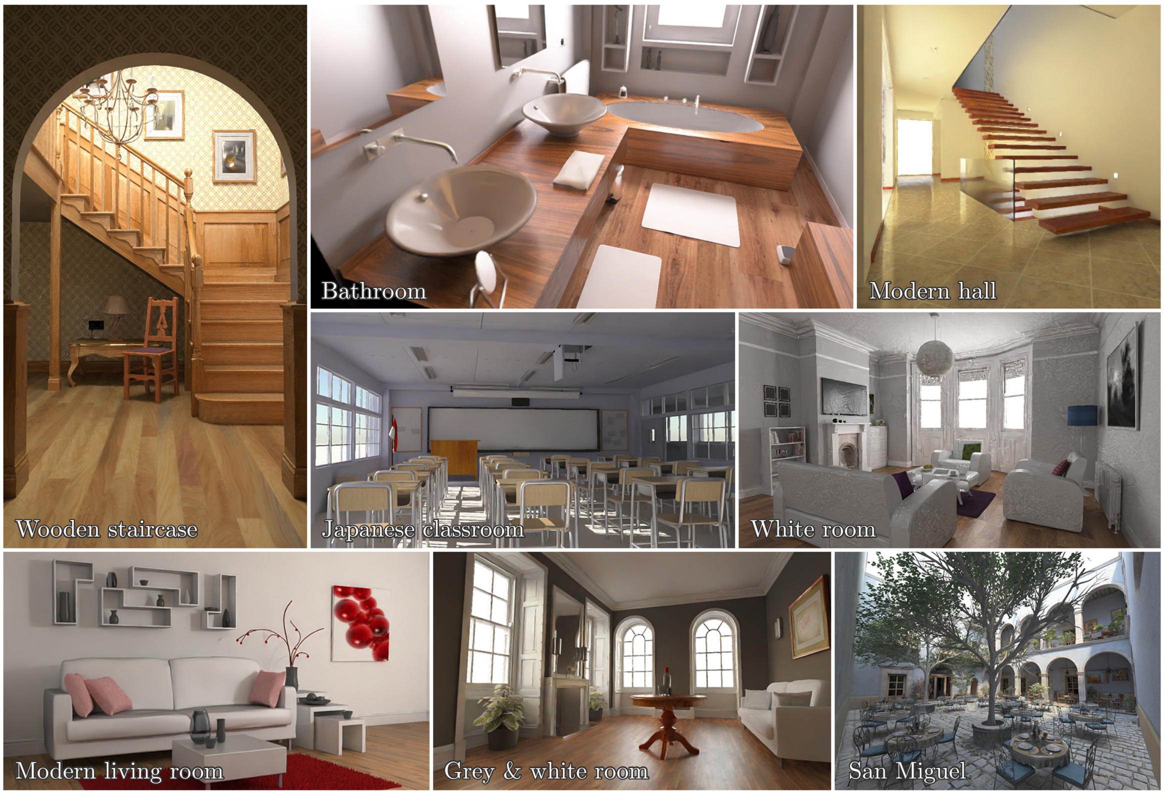


FIGURE 6.13: Collage of the surrogates used in our experiments, obtained by denoising the input estimates using Intel Open Image Denoise [116].

Progressive rendering. Our optimisation methods produce biased pixel estimates through manipulating the input samples; this is true even for a-priori methods where the sampling is completely deterministic. Nevertheless, consistency can be achieved through a simple progressive-rendering scheme: For each pixel, newly generated samples are cumulatively averaged into a fixed set of per-pixel estimates that are periodically passed to the optimisation to obtain an updated image. Each individual estimate will converge to the true pixel value, thus the optimised image will also approach the ground truth—with bounded memory footprint. Interestingly, convergence is guaranteed regardless of the optimisation method and surrogate used, though better methods and surrogates will yield better starting points. Lastly, adaptive sampling is naturally supported by vertical methods as they are agnostic of differences in sample counts between pixels.

6.9 Conclusion

We devise a formal treatment of image-space error distribution in Monte Carlo rendering from both quantitative and perceptual aspects. Our formulation bridges the gap between halftoning and rendering by interpreting the error distribution problem as an extension of non-uniform multi-tone energy minimisation halftoning. To guide the distribution of rendering error, we employ a perceptual kernel-based model whose practical optimisation can deliver improvements not achievable by prior methods given the same sampling data. Our model provides valuable insights as well as a framework to further study the problem and its solutions.

A promising avenue for future research is to adapt even stronger perceptual error models. Prior work has already demonstrated a strong potential in reducing Monte Carlo noise visibility error using visual masking [34, 197]. Robust metrics, other than squared $\mathcal{L}_2$ norm, can also be considered with possible nonlinear relationships.

Our framework could conceivably be extended beyond the human visual system, i.e., for optimising the inputs to other types of image processing such as denoising. For such tasks, one could consider lifting the assumption of a fixed kernel to obtain an even more general problem where the kernel and sample distribution are optimised simultaneously (or alternatingly).

Chapter 7
Conclusion and Outlook

7.1 Conclusion

In the current thesis we have developed efficient strategies for sparse PDE-based image reconstruction. We have improved upon prior approaches both in terms of quality and efficiency. Our feature inpainting framework from Chapter 2 allows us to achieve MSE improvements of more than 60% while using the same total mask density. We have seen that if the features are selected to complement for the inpainting operator's weaknesses they can even suppress artifacts such as the logarithmic singularities inherent to harmonic inpainting in 2D. This is achieved thanks to our spatial optimisation algorithm from Chapter 4, which automatically allocates the percentage of relevant features by adapting to the image and inpainting operator. In the setting of harmonic inpainting we see that our spatial optimisation replaces most of the pointwise interpolation constraints by local integral constraints. The latter circumvents the logarithmic singularities problem, since interpolation of local integrals is well-defined even in the continuous setting, unlike interpolation of pointwise values for $H^1(\Omega)$ "functions" in 2D. Our spatial optimisation provides further insights, as it allocates few derivative features to EED inpainting. The reason for this stems from the fact that EED already has a mechanism for inferring edges. This emphasises the principle that **one should choose features that complement the inpainting operator**.

A major obstacle to the widespread adoption of PDE-based reconstruction methods has been their high computational cost and limited scalability – especially when compared to fast, transform-based codecs like JPEG and JPEG2000. For nearly orthogonal or unitary transforms, even spatial and tonal optimisation becomes trivial, as the best m-term approximation can be obtained simply by thresholding the magnitudes of the transformed coefficients. This simplicity does not extend to classical PDE-based methods such as harmonic and EED inpainting. In these cases, the associated matrices are far from orthogonal and often exhibit very large condition numbers. As a result, analytic strategies like the one proposed by Belhachmi et al. [28] must rely on error diffusion rather than straightforward thresholding, yet still result in masks that are far from optimal.

To address these challenges, we have placed a strong emphasis on efficiency throughout our work, particularly in Chapter 3 and Chapter 4 of this thesis. On the CPU, we reformulated harmonic inpainting within a finite element method (FEM) framework and employed adaptive Delaunay triangulations. This approach reduced both the condition number and the number of unknowns, enabling linear runtime scaling of both inpainting and data optimisation with respect to image resolution. As an unexpected by-product, the adaptive mesh also allowed us to suppress the logarithmic singularities characteristic of harmonic inpainting. On the GPU, we achieved real-time performance for 4K image inpainting by carefully integrating multigrid methods, domain decomposition, and Krylov subspace solvers. Additionally, we explored several strategies to enhance the efficiency of tonal optimisation.

The development of efficient and high quality algorithms for inpainting enabled us, in Chapter 5, to shift our focus towards exploring the relationship between denoising and inpainting. Rather surprisingly, we have seen that the two are tightly linked, and we could establish a relationship between the mask density in homogeneous diffusion inpainting, and the diffusion time in homogeneous diffusion denoising. Our formulation relating the two worlds was based on an interpretation of the averaging of multiple inpaintings as an estimator for a denoising process. We provided a comprehensive probabilistic and deterministic theory for the latter, that was not tied to a specific inpainting or mask generation process.

Finally, in Chapter 6, we developed a theoretical framework for perceptual error optimisation in Monte Carlo rendering. Our work provides a solid foundation for the previously heuristic motivation behind favouring blue noise error distributions. Blue noise is desirable because it decays rapidly under low-pass filtering, making it less perceptible in rendered images. Our framework allowed us to generalise a number of classical halftoning algorithms such as dithering, error-diffusion, and iterative energy minimisation halftoning, to the Monte Carlo rendering setting.

7.2 Outlook

While our work addresses several practical challenges and theoretical questions, it also offers promising directions for future research. For instance, in our feature inpainting framework we did not explore the inequality-constrained case. The theoretical extension is relatively straightforward – it is sufficient to consider the Karush-Kuhn-Tucker (KKT) conditions for the inequality-constrained optimisation problem. The main difficulty lies in developing efficient algorithms to solve the resulting problem in the inpainting setting, since the latter results in very large and sparse problems. For the equality-constrained case, we proposed an SQP-inspired algorithm. This naturally suggests using an inequality-constrained SQP-inspired approach in the inequality-constrained setting. Interior point methods could also be explored as a viable alternative.

Another open problem is identifying practically useful nonlinear features. While incorporating nonlinear inpainting into our feature framework led to substantial qualitative improvements, simply replacing linear features with their nonlinear counterparts did not result in better reconstructions. In the context of inequality-constrained inpainting, however, a promising idea for feature design emerges: we could enforce local minima and maxima in order to prevent overshoots and undershoots. This could be beneficial for operators like biharmonic inpainting.

In our FEM inpainting experiments we observed that extensions to the biharmonic setting did not yield benefits. A natural next step would be to explore FEM EED inpainting. A more ambitious direction would be to move towards the reconstruction of surfaces from sparse data as in the work of Bae and Weickert [19], which is a more natural application area for FEM. In fact, adapting the methods developed in this thesis to surface settings is relatively straightforward. This primarily involves replacing standard differential operators with their differential geometric counterparts on manifolds. For example, the 5-point stencil Laplacian can be replaced by the cotangent Laplacian, which discretises the Laplace-Beltrami operator on triangular meshes. Naturally, this transition introduces new challenges – it makes harmonic inpainting nonlinear in the surface setting, which requires algorithmic adaptations.

One notable limitation of our efficient inpainting methods is that they were specifically tailored to harmonic inpainting. However, techniques such as multigrid and domain decomposition are not inherently restricted to this setting – they can be adapted to polyharmonic inpainting, and with more effort, potentially to other nonlinear models such as EED. This represents an important direction for future research, especially given that EED significantly outperforms harmonic inpainting in terms of reconstruction quality.

Although our theory of denoising via inpainting is quite comprehensive, the connection between diffusion and inpainting in the 2D setting was demonstrated only empirically. A natural and important extension of this work would be to develop a rigorous theoretical proof of this relationship in two dimensions.

Last but not least, we note that our framework for perceptual error optimisation in Monte Carlo rendering has already served as a base for other works. The most closely related work extends our framework to the spatio-temporal setting [138].

List of Figures

List of Tables

Bibliography

[1] R. Achanta, N. Arvanitopoulos, and S. Süsstrunk. Extreme image completion. In *Proc. 42nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1333–1337, New Orleans, Mar. 2017.

[2] R. D. Adam, P. Peter, and J. Weickert. Denoising by inpainting. In F. Lauze, Y. Dong, and A. B. Dahl, editors, *Scale Space and Variational Methods in Computer Vision*, volume 10302 of *Lecture Notes in Computer Science*, pages 121–132. Springer, Cham, 2017.

[3] M. D. Adams. A highly-effective incremental/decremental Delaunay mesh-generation strategy for image representation. *Signal Processing*, 93(4):749–764, 2013.

[4] C. Aguerrebere, A. Almansa, J. Delon, Y. Gousseau, and P. Musé. A Bayesian hyperprior approach for joint image denoising and interpolation, with an application to HDR imaging. *IEEE Transactions on Computational Imaging*, 3(4):633–646, 2017.

[5] A. G. M. Ahmed and P. Wonka. Screen-space blue-noise diffusion of Monte Carlo sampling error via hierarchical ordering of pixels. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 39(6), 2020.

[6] N. Ahmed, T. Natarajan, and K. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[7] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. KAZE features. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision – ECCV 2012*, volume 7577 of *Lecture Notes in Computer Science*, pages 214–227. Springer, Berlin, 2012.

[8] J. P. Allebach and B. Liu. Random quasi-periodic halftone process. *Journal of the Optical Society of America*, 66(9):909–917, Sept. 1976.

[9] T. Alt, P. Peter, and J. Weickert. Learning sparse masks for diffusion-based image inpainting. In A. J. Pinho, P. Georgieva, L. F. Teixeira, and J. A. Sánchez, editors, *Pattern Recognition and Image Analysis*, volume 13256 of *Lecture Notes in Computer Science*, pages 528–539. Springer, Cham, 2022.

[10] T. Alt, K. Schrader, M. Augustin, P. Peter, and J. Weickert. Connections between numerical algorithms for PDEs and neural networks. *Journal of Mathematical Imaging and Vision*, 65:185–208, 2023.

[11] M. Analoui and J. P. Allebach. Model-based halftoning using direct binary search. In B. E. Rogowitz, editor, *Human Vision, Visual Processing, and Digital Display III*, volume 1666, pages 96–108. SPIE Press, 1992.

[12] D. Anastassiou. Error diffusion coding for A/D conversion. *IEEE Transactions on Circuits and Systems*, 36(9):1175–1186, 1989.

[13] S. Andris, P. Peter, and J. Weickert. A proof-of-concept framework for PDE-based video compression. In *Proc. 32nd Picture Coding Symposium*, Nuremberg, Germany, Dec. 2016.

[14] S. Andris, P. Peter, R. M. Kaja Mohideen, J. Weickert, and S. Hoffmann. Inpainting-based video compression in FullHD. In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon, editors, *Scale Space and Variational Methods in Computer Vision*, volume 12679 of *Lecture Notes in Computer Science*, pages 425–436. Springer, Cham, 2021.

[15] W. Arrighetti. The academy color encoding system (ACES): A professional color-management framework for production, post-production and archival of still and motion pictures. *Journal of Imaging*, 3:40, Sept. 2017.

[16] M. Augustin, J. Weickert, and S. Andris. Pseudodifferential inpainting: The missing link between PDE- and RBF-based interpolation. In J. Lellmann, M. Burger, and J. Modersitzki, editors, *Scale Space and Variational Methods in Computer Vision*, volume 11603 of *Lecture Notes in Computer Science*, pages 67–78. Springer, Cham, 2019.

[17] J. Aujol, S. Ladjal, and S. Masnou. Exemplar-based inpainting from a variational point of view. *SIAM Journal on Applied Mathematics*, 42(3):1246–1285, 2010.

[18] F. Aurenhammer, R. Klein, and D. Lee. *Voronoi Diagrams And Delaunay Triangulations.* World Scientific, Singapore, 2013.

[19] 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, editors, *Mathematical Methods for Curves and Surfaces*, volume 5862 of *Lecture Notes in Computer Science*, pages 1–14. Springer, Berlin, 2010.

[20] E. Bänsch and K. Mikula. A coarsening finite element strategy in image selective smoothing. *Computation and Visualization in Science*, 1:53–61, 1997.

[21] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics*, 28(3):1–11, July 2009.

[22] P. G. Barten. *Contrast Sensitivity of the Human Eye and its Effects on Image Quality.* SPIE Press, 1999.

[23] B. E. Bayer. An optimum method for two-level rendition of continuous-tone pictures. In *Proceedings of IEEE International Conference on Communications*, volume 26, pages 11–15, 1973.

[24] L. Belcour and E. Heitz. Lessons learned and improvements when building screen-space samplers with blue-noise error distribution. In *ACM SIGGRAPH 2021 Talks*, SIGGRAPH '21, New York, NY, 2021.

[25] Z. Belhachmi and T. Jacumin. Optimal interpolation data for PDE-based compression of images with noise. *Communications in Nonlinear Science and Numerical Simulation*, 109:Article 106278, 2022.

[26] Z. Belhachmi and T. Jacumin. Iterative approach to image compression with noise: Optimizing spatial and tonal data, 2022. URL `https://arxiv.org/abs/2209.14706`.

[27] Z. Belhachmi and T. Jacumin. Adjoint method in PDE-based image compression, 2023. URL `https://arxiv.org/abs/2302.02665`.

[28] Z. Belhachmi, D. Bucur, B. Burgeth, and J. Weickert. How to choose interpolation data in images. *SIAM Journal on Applied Mathematics*, 70(1):333–352, June 2009.

[29] J. Berger and G. T. Lasher. The use of discrete Green's functions in the numerical solution of Poisson's equation. *Illinois Journal of Mathematics*, 2:593–607, 1958.

[30] M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. SIGGRAPH 2000*, pages 417–424, New Orleans, July 2000.

[31] M. Bertalmío, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Transactions on Image Processing*, 12(8):882–889, Aug. 2003.

[32] Å. Björk. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

[33] M. R. Bolin and G. W. Meyer. A frequency based ray tracer. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 409–418, New York, 1995.

[34] M. R. Bolin and G. W. Meyer. A perceptually based adaptive sampling algorithm. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 299–309, New York, 1998.

[35] S. Bonettini, I. Loris, F. Porta, M. Prato, and S. Rebegoldi. On the convergence of a linesearch based proximal-gradient method for nonconvex optimization. *Inverse Problems*, 33(5):Article 055005, Mar. 2017.

[36] F. Bornemann and P. Deuflhard. The cascadic multigrid method for elliptic problems. *Numerische Mathematik*, 75:135–152, 1996.

[37] S. Boujena, K. Bellaj, O. Gouasnouane, and E. Guarmah. An improved nonlinear model for image inpainting. *Applied Mathematical Sciences*, 9(124):6189–6205, Jan. 2015.

[38] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31(138):333–390, Apr. 1977.

[39] M. Breuß, L. Hoeltgen, and G. Radow. Towards PDE-based video compression with optimal masks prolongated by optic flow. *Journal of Mathematical Imaging and Vision*, 63(2):144–156, Feb. 2021.

[40] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial.* SIAM, Philadelphia, second edition, 2000.

[41] E.-M. Brinkmann, M. Burger, and I. Grah. Regularization with sparse vector fields: From image compression to TV-type reconstruction. In J. Aujol, M. Nikolova, and N. Papadakis, editors, *Scale Space and Variational Methods in Computer Vision*, volume 9087 of *Lecture Notes in Computer Science*, pages 191–202. Springer, Berlin, 2015.

[42] A. Bruhn and J. Weickert. A confidence measure for variational optic flow methods. In R. Klette, R. Kozera, L. Noakes, and J. Weickert, editors, *Geometric Properties from Incomplete Data*, volume 31 of *Computational Imaging and Vision*, pages 283–297. Springer, Dordrecht, 2006.

[43] A. Buades, B. Coll, and J.-M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation*, 4(2):490–530, 2005.

[44] M. Burger, L. He, and C. Schönlieb. Cahn–Hilliard inpainting and a generalization for grayvalue images. *SIAM Journal on Imaging Sciences*, 2(4):1129–1167, 2009.

[45] R. H. Byrd, F. E. Curtis, and J. Nocedal. An inexact SQP method for equality constrained optimization. *SIAM Journal on Optimization*, 19(1):351–369, 2008.

[46] S. Carlsson. Sketch based coding of grey level images. *Signal Processing*, 15:57–83, 1988.

[47] V. Caselles, B. Coll, and J. Morel. Junction detection and filtering. In F. Cucker and M. Shub, editors, *Foundations of Computational Mathematics*, pages 23–42. Springer, Berlin, 1997.

[48] A. Celarek, W. Jakob, M. Wimmer, and J. Lehtinen. Quantifying the error of light transport algorithms. *Computer Graphics Forum*, 38(4):111–121, 2019.

[49] C. R. A. Chaitanya, A. S. Kaplanyan, C. Schied, M. Salvi, A. Lefohn, D. Nowrouzezahrai, and T. Aila. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics*, 36(4):1–12, July 2017.

[50] R. H. Chan, C. Ho, and M. Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on Image Processing*, 14(10):1479–1485, 2005.

[51] T. F. Chan and J. Shen. Non-texture inpainting by curvature-driven diffusions (CDD). *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.

[52] R. Chandra, S. C. Eisenstat, and M. H. Schultz. The modified Conjugate Residual method for partial differential equations. Technical report, Yale University, New Haven, CT, May 1977.

[53] J. Chang, B. Alain, and V. Ostromoukhov. Structure-aware error diffusion. *ACM Transactions on Graphics*, 28(5):1–8, Dec. 2009.

[54] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on Image Processing*, 6(2):298–311, 1997.

[55] R. Chen, J. Huang, and X.-C. Cai. A parallel domain decomposition algorithm for large scale image denoising. *Inverse Problems and Imaging*, 13(6):1259–1282, Dec. 2019.

[56] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[57] Y. Chen, R. Ranftl, and T. Pock. A bi-level view of inpainting-based image compression. In Z. Kúkelová and J. Heller, editors, *Proc. 19th Computer Vision Winter Workshop*, pages 19–25, Křtiny, Czech Republic, Feb. 2014.

[58] V. Chizhov and J. Weickert. Efficient data optimisation for harmonic inpainting with finite elements. In N. Tsapatsoulis, A. Panayides, T. Theo, A. Lanitis, C. Pattichis, and M. Vento, editors, *Computer Analysis of Images and Patterns*, volume 13053 of *Lecture Notes in Computer Science*, pages 432–441. Springer, Cham, 2021.

[59] V. Chizhov, I. Georgiev, K. Myszkowski, and G. Singh. Perceptual error optimization for Monte Carlo rendering. *ACM Transactions on Graphics*, 41(3), Mar. 2022.

[60] F. Chung and S. Yau. Discrete Green's functions. *Journal of Combinatorial Theory, Series A*, 91(12):191–214, 2000.

[61] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. SIAM, Philadelphia, 2000.

[62] P. Craven and G. Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403, Jan. 1978.

[63] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13(9): 1200–1212, Sept. 2004.

[64] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, Aug. 2007.

[65] Q. Dai, H. Chopp, E. Pouyet, O. Cossairt, M. Walton, and A. K. Katsaggelos. Adaptive image sampling using deep learning and its application on X-ray fluorescence image reconstruction. *IEEE Transactions on Multimedia*, 22(10):2564–2578, 2020.

[66] S. J. Daly. Subroutine for the generation of a two dimensional human visual contrast sensitivity function. Technical Report 233203Y, Eastman Kodak: Rochester, NY, 1987.

[67] S. J. Daly. Visible differences predictor: an algorithm for the assessment of image fidelity. In B. E. Rogowitz, editor, *Human Vision, Visual Processing, and Digital Display III*, volume 1666, pages 2–15. SPIE Press, 1992.

[68] V. Daropoulos, M. Augustin, and J. Weickert. Sparse inpainting with smoothed particle hydrodynamics. *SIAM Journal on Imaging Sciences*, 14(4):1669–1704, Nov. 2021.

[69] R. J. Deeley, N. Drasdo, and W. N. Charman. A simple parametric model of the human ocular modulation transfer function. *Ophthalmic and Physiological Optics*, 11(1):91–93, 1991.

[70] L. Demaret, N. Dyn, and A. Iske. Image compression by linear splines over adaptive triangulations. *Signal Processing*, 86(7):1604–1616, 2006.

[71] M. Di Martino and G. Facciolo. An analysis and implementation of multigrid Poisson solvers with verified linear complexity. *Image Processing On Line*, 8: 192–218, 2018.

[72] R. Distasi, M. Nappi, and S. Vitulano. Image compression by B-tree triangular coding. *IEEE Transactions on Communications*, 45(9):1095–1100, Sept. 1997.

[73] V. Dolean, P. Jolivet, and F. Nataf. *An Introduction to Domain Decomposition Methods: Algorithms, Theory, and Parallel Implementation.* SIAM, Philadelphia, 2015.

[74] B. Dong, Q. Jiang, and Z. Shen. Image restoration: Wavelet frame shrinkage, nonlinear evolution PDEs, and beyond. *Multiscale Modeling and Simulation*, 15 (1):606–660, 2017.

[75] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrica*, 81(3):425–455, 1994.

[76] J. Duchon. Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. *RAIRO Analyse Numérique*, 10:5–12, 1976.

[77] S. Durand and M. Nikolova. Stability of image restoration by minimizing regularized objective functions. In *Proc. First IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 73–80, Vancouver, Canada, July 2001.

[78] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proc. Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038, Corfu, Greece, Sept. 1999.

[79] S. C. Eisenstat and H. F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[80] B. E. El Marzouki and M. D. Adams. An improved incremental/decremental Delaunay mesh-generation strategy for image representation. In *2017 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 1–6, Victoria, Canada, Aug. 2017.

[81] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer, New York, 2010.

[82] M. Elad, B. Kawar, and G. Vaksman. Image denoising: The deep learning revolution and beyond — a survey paper. *SIAM Journal on Imaging Sciences*, 16 (3):1594–1654, 2023.

[83] 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, editors, *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, volume 5681 of *Lecture Notes in Computer Science*, pages 331–344. Springer, Berlin, 2009.

[84] J. A. Ferwerda, S. N. Pattanaik, P. Shirley, and D. P. Greenberg. A model of visual adaptation for realistic image synthesis. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 249—258, New York, 1996.

[85] R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial grey scale. *Proceedings of the Society of Information Display*, 17:75–77, 1976.

[86] D. C.-L. Fong and M. A. Saunders. CG versus MINRES: An empirical comparison. *SQU Journal for Science*, 17(1):44–62, 2012.

[87] D. S. Fritsch. A medial description of greyscale image structure by gradient-limited diffusion. In R. A. Robb, editor, *Visualization in Biomedical Computing '92*, volume 1808 of *Proceedings of SPIE*, pages 105–117. SPIE Press, Bellingham, 1992.

[88] D. Gaa, V. Chizhov, P. Peter, J. Weickert, and R. D. Adam. Connecting image inpainting with denoising in the homogeneous diffusion setting. *Advances in Continuous and Discrete Models*, 74, 2025.

[89] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, and H.-P. Seidel. Image compression with anisotropic diffusion. *Journal of Mathematical Imaging and Vision*, 31(2–3):255–269, July 2008.

[90] J. Gautier, O. Le Meur, and C. Guillemot. Efficient depth map compression based on lossless edge coding and diffusion. In *Proc. 2012 Picture Coding Symposium*, pages 81–84, Kraków, Poland, May 2012.

[91] F. Gazzola, H.-C. Grunau, and G. Sweers. *Polyharmonic Boundary Value Problems*, volume 1991 of *Lecture Notes in Mathematics.* Springer, Heidelberg, first edition, June 2010.

[92] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[93] I. Georgiev and M. Fajardo. Blue-noise dithered sampling. In *ACM SIGGRAPH 2016 Talks*, SIGGRAPH '16, New York, 2016.

[94] A. J. González, J. Bacca, G. R. Arce, and D. L. Lau. Alpha stable human visual system models for digital halftoning. In B. E. Rogowitz, T. N. Pappas, and S. J. Daly, editors, *Human Vision and Electronic Imaging XI*, volume 6057, pages 180–191. SPIE Press, 2006.

[95] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73(2):325–348, 1987.

[96] C. Guillemot and O. Le Meur. Image inpainting: Overview and recent advances. *IEEE Signal Processing Magazine*, 31(1):127–144, 2014.

[97] O. G. Guleryuz. Iterated denoising for image recovery. In *Proc. 2002 Data Compression Conference*, pages 3–12, Snowbird, UT, 2002.

[98] W. Hackbusch. *Multigrid Methods and Applications*. Springer, New York, 1985.

[99] A. Hamza, H. Krim, and G. Unal. Unifying probabilistic and variational estimation. *IEEE Signal Processing Magazine*, 19(5):37–47, 2002.

[100] R. L. Harder and R. N. Desmarais. Interpolation using surface splines. *Journal of Aircraft*, 9(2):189–191, 1972.

[101] K. Hayami. Convergence of the Conjugate Gradient method on singular systems, 2020. URL `https://arxiv.org/abs/1809.00793`.

[102] P. Heid and T. P. Wihler. A modified Kačanov iteration scheme with application to quasilinear diffusion models. *ESAIM: Mathematical Modelling and Numerical Analysis*, 56(2):433–450, 2022.

[103] E. Heitz and L. Belcour. Distributing Monte Carlo errors as a blue noise in screen space by permuting pixel seeds between frames. *Computer Graphics Forum*, 38(4): 149–158, 2019.

[104] E. Heitz, L. Belcour, V. Ostromoukhov, D. Coeurjolly, and J.-C. Iehl. A low-discrepancy sampler that distributes Monte Carlo errors as a blue noise in screen space. In *ACM SIGGRAPH 2019 Talks*, SIGGRAPH '19, New York, 2019.

[105] M. Hestenes and E. Stiefel. Method of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–438, 1951.

[106] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, second edition, 2002.

[107] S. Hocevar and G. Niger. RETRACTED: Reinstating Floyd-Steinberg: Improved metrics for quality assessment of error diffusion algorithms. In A. Elmoataz, O. Lezoray, F. Nouboud, and D. Mammass, editors, *Image and Signal Processing*, pages 38–45. Springer, Berlin, 2008.

[108] L. Hoeltgen and J. Weickert. Why does non-binary mask optimisation work for diffusion-based image compression? In X.-C. Tai, E. Bae, T. F. Chan, S. Y. Leung, and M. Lysaker, editors, *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, volume 8932 of *Lecture Notes in Computer Science*, pages 85–98. Springer, Berlin, 2015.

[109] L. Hoeltgen, S. Setzer, and J. Weickert. An optimal control approach to find sparse data for Laplace interpolation. In A. Heyden, F. Kahl, C. Olsson, M. Oskarsson, and X.-C. Tai, editors, *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, volume 8081 of *Lecture Notes in Computer Science*, pages 151–164. Springer, Berlin, 2013.

[110] L. Hoeltgen, M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, S. Setzer, D. Johannsen, F. Neumann, and B. Doerr. Optimising spatial and tonal data for PDE-based inpainting. In M. Bergounioux, G. Peyré, C. Schnörr, J.-P. Caillau, and T. Haberkorn, editors, *Variational Methods in Imaging and Geometric Control*, pages 35–83. De Gruyter, Berlin, 2017.

[111] S. Hoffmann. *Competitive Image Compression with Linear PDEs.* PhD thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, Dec. 2016.

[112] S. Hoffmann, M. Mainberger, J. Weickert, and M. Puhl. Compression of depth maps with segment-based homogeneous diffusion. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 319–330. Springer, Berlin, 2013.

[113] S. Hoffmann, G. Plonka, and J. Weickert. Discrete Green's functions for harmonic and biharmonic inpainting with sparse atoms. In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 8932 of *Lecture Notes in Computer Science*, pages 169–182. Springer, Berlin, 2015.

[114] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17: 185–203, 1981.

[115] T. Iijima. Basic theory on normalization of pattern (in case of typical one-dimensional pattern). *Bulletin of the Electrotechnical Laboratory*, 26:368–388, 1962. In Japanese.

[116] Intel. Intel open image denoise, 2018. URL `https://www.openimagedenoise.org`.

[117] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Dover, New York, 2009.

[118] F. Jost, P. Peter, and J. Weickert. Compressing flow fields with edge-aware homogeneous diffusion inpainting. In *Proc. 45th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2198–2202, Barcelona, Spain, May 2020.

[119] F. Jost, P. Peter, and J. Weickert. Compressing piecewise smooth images with the Mumford-Shah cartoon model. In *Proc. 28th European Signal Processing Conference (EUSIPCO)*, pages 511–515, Amsterdam, Netherlands, Jan. 2021.

[120] F. Jost, V. Chizhov, and J. Weickert. Optimising different feature types for inpainting-based image representations. In *Proc. 48th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes, Greece, June 2023.

[121] I. Jumakulyyev and T. Schultz. Fourth-order anisotropic diffusion for inpainting and image compression. In E. Özarslan, T. Schultz, E. Zhang, and A. Fuster, editors, *Anisotropy Across Fields and Scales*, pages 99–124. Springer, Cham, 2021.

[122] I. Jumakulyyev and T. Schultz. Combining image space and q-space PDEs for lossless compression of diffusion MR images. *Journal of Mathematical Imaging and Vision*, 65(4):644–656, Aug. 2023.

[123] L. M. Kachanov. Variational methods of solution of plasticity problems. *Journal of Applied Mathematics and Mechanics*, 23(3):880–883, 1959.

[124] J. T. Kajiya. The rendering equation. *SIGGRAPH Computer Graphics*, 20(4): 143–150, Aug. 1986.

[125] E. M. Kalmoun and M. M. S. Nasser. Harmonic image inpainting using the charge simulation method. *Pattern Analysis and Applications*, 25:795–806, 2022.

[126] N. Kämper and J. Weickert. Domain decomposition algorithms for real-time homogeneous diffusion inpainting in 4K. In *Proc. 47th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1680–1684. IEEE Computer Society Press, Singapore, May 2022.

[127] N. Kämper, V. Chizhov, and J. Weickert. Efficient parallel algorithms for inpainting-based representations of 4K images – part I: Homogeneous diffusion inpainting, 2024. URL `https://arxiv.org/abs/2401.06744`.

[128] N. Kämper, V. Chizhov, and J. Weickert. Efficient parallel data optimization for homogeneous diffusion inpainting of 4K images. *SIAM Journal on Imaging Sciences*, 18(1):701–734, 2025.

[129] F. M. W. Kanters, M. Lillholm, R. Duits, B. J. P. Jansen, B. Platel, L. M. J. Florack, and B. M. ter Haar Romeny. On image reconstruction from multiscale top points. In R. Kimmel, N. Sochen, and J. Weickert, editors, *Scale Space and PDE Methods in Computer Vision*, volume 3459 of *Lecture Notes in Computer Science*, pages 431–439. Springer, Berlin, 2005.

[130] A. S. Kaplanyan, A. Sochenov, T. Leimkühler, M. Okunev, T. Goodall, and G. Rufo. Deepfovea: Neural reconstruction for foveated rendering and video compression using learned statistics of natural videos. *ACM Transactions on Graphics*, 38(6), Nov. 2019.

[131] L. Karos, P. Bheed, P. Peter, and J. Weickert. Optimising data for exemplar-based inpainting. In J. Blanc-Talon, D. Helbert, W. Philips, D. Popescu, and P. Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 11182 of *Lecture Notes in Computer Science*, pages 547–558. Springer, Berlin, 2018.

[132] M. Katsurada. A mathematical study of the charge simulation method by use of peripheral conformal mappings. *Memoirs of the Institute of Sciences and Technology, Meiji University*, 37:195–212, Nov. 1999.

[133] J. Kačur and K. Mikula. Solution of nonlinear diffusion appearing in image smoothing and edge detection. *Applied Numerical Mathematics*, 17:47–59, 1995.

[134] E. J. King, G. Kutyniok, and W.-Q. Lim. Image inpainting: Theoretical analysis and comparison of algorithms. In D. Van De Ville, V. K. Goyal, and M. Papadakis, editors, *Wavelets and Sparsity XV*, volume 8858 of *Proceedings of SPIE*. SPIE Press, Bellingham, 2013.

[135] H. Koge, Y. Ito, and K. Nakano. A GPU implementation of clipping-free halftoning using the direct binary search. In X.-h. Sun, W. Qu, I. Stojmenovic, W. Zhou, Z. Li, H. Guo, G. Min, T. Yang, Y. Wu, and L. Liu, editors, *Algorithms and Architectures for Parallel Processing*, pages 57–70. Springer, Cham, 2014.

[136] T. Kohlberger. *Variational Domain Decomposition for Parallel Image Processing*. PhD thesis, School of Business Informatics and Mathematics, University of Mannheim, Mannheim, Germany, 2007.

[137] T. Kohlberger, C. Schnörr, A. Bruhn, and J. Weickert. Domain decomposition for variational optical flow computation. *IEEE Transactions on Image Processing*, 14 (8):1125–1137, Aug. 2005.

[138] M. Korać, C. Salaün, I. Georgiev, P. Grittmann, P. Slusallek, K. Myszkowski, and G. Singh. Perceptual error optimization for Monte Carlo animation rendering. In *SIGGRAPH Asia 2023 Conference Papers*, SA '23, New York, 2023.

[139] H. Köstler, M. Stürmer, C. Freundl, and U. Rüde. PDE based video compression in real time. Technical Report 07-11, Lehrstuhl für Informatik 10, Univ. Erlangen–Nürnberg, Germany, 2007.

[140] L. Kuipers and H. Niederreiter. *Uniform Distribution of Sequences*. Dover, New York, 2005.

[141] V. Kupradze and M. Aleksidze. The method of functional equations for the approximate solution of certain boundary value problems. *USSR Computational Mathematics and Mathematical Physics*, 4(4):82–126, 1964.

[142] A. Kuznetsov, N. K. Kalantari, and R. Ramamoorthi. Deep adaptive sampling for low sample count rendering. *Computer Graphics Forum*, 37(4):35–44, 2018.

[143] M. Lalee, J. Nocedal, and T. Plantenga. On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM Journal on Optimization*, 8(3):682–706, 1998.

[144] M. G. Larson and F. Bengzon. *The Finite Element Method: Theory, Implementation, and Applications.* Texts in Computational Science and Engineering. Springer, Berlin, 2013.

[145] E. G. Larsson and Y. Selen. Linear regression with a sparse parameter vector. *IEEE Transactions on Signal Processing*, 55(2):451–460, 2007.

[146] D. L. Lau and G. R. Arce. *Modern Digital Halftoning.* CRC Press, Boca Raton, second edition, 2007.

[147] R. Laumont, V. D. Bortoli, A. Almansa, J. Delon, A. Durmus, and M. Pereyra. Bayesian imaging using Plug & Play priors: When Langevin meets Tweedie. *SIAM Journal on Imaging Sciences*, 15(2):701–737, 2022.

[148] M. Lebrun, M. Colom, A. Buades, and J. M. Morel. Secrets of image denoising cuisine. *Acta Numerica*, 21(1):475–576, 2012.

[149] Y. Li, M. Sjostrom, U. Jennehag, and R. Olsson. A scalable coding approach for high quality depth image compression. In *Proc. 3DTV-Conference: The True Vision – Capture, Transmission and Display of 3D Video*, Zurich, Switzerland, Oct. 2012.

[150] A. Lim, Y. Liu, and F. Roosta. Conjugate Direction methods under inconsistent systems, 2024. URL `https://arxiv.org/abs/2401.11714`.

[151] T. Lindeberg. Scale-space for discrete signals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3):234–254, 1990.

[152] T. Lindeberg. *Scale-Space Theory in Computer Vision.* Kluwer, Boston, 1994.

[153] D. L. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[154] J. Lubin. A visual discrimination model for imaging system design and evaluation. In E. Peli, editor, *Vision Models for Target Detection and Recognition*, pages 245–283. World Scientific, 1995.

[155] M. Mainberger, A. Bruhn, J. Weickert, and S. Forchhammer. Edge-based compression of cartoon-like images with homogeneous diffusion. *Pattern Recognition*, 44 (9):1859–1873, Sept. 2011.

[156] 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, editors, *Scale Space and Variational Methods in Computer Vision*, volume 6667 of *Lecture Notes in Computer Science*, pages 26–37. Springer, Berlin, 2011.

[157] S. Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, San Diego, second edition, 1999.

[158] S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[159] J. L. Mannos and D. J. Sakrison. The effects of a visual fidelity criterion of the encoding of images. *IEEE Transactions on Information Theory*, 20(4):525–536, 1974.

[160] R. Mantiuk, S. J. Daly, K. Myszkowski, and H.-P. Seidel. Predicting visible differences in high dynamic range images: model and its calibration. In B. E. Rogowitz, T. N. Pappas, and S. J. Daly, editors, *Human Vision and Electronic Imaging X*, volume 5666, pages 204–214. SPIE Press, 2005.

[161] R. Mantiuk, K. J. Kim, A. G. Rempel, and W. Heidrich. HDR-VDP-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Transactions on Graphics*, 30(4), July 2011.

[162] D. Marwood, P. Massimino, M. Covell, and S. Baluja. Representing images in 200 bytes: Compression via triangulation. In *Proc. 2018 IEEE International Conference on Image Processing (ICIP)*, pages 405–409, Athens, Greece, Oct. 2018.

[163] S. Masnou and J.-M. Morel. Level lines based disocclusion. In *Proc. 1998 IEEE International Conference on Image Processing*, volume 3, pages 259–263, Chicago, IL, Oct. 1998.

[164] A. McAdams, E. Sifakis, and J. Teran. A parallel multigrid Poisson solver for fluids simulation on large grids. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 65–74, Goslar, Germany, 2010.

[165] J. Meinguet. Multivariate interpolation at arbitrary points made simple. *Journal of Applied Mathematics and Physics (ZAMP)*, 30:292–304, Mar. 1979.

[166] Y. Melnikov and M. Melnikov. Computability of series representations for Green's functions in a rectangle. *Engineering Analysis with Boundary Elements*, 30(9): 774–780, 2006.

[167] P. T. Metaxas. Parallel digital halftoning by error-diffusion. In *Proceedings of the Paris C. Kanellakis Memorial Workshop on Principles of Computing & Knowledge*, pages 35–41, 2003.

[168] A. R. Mitchell and D. F. Griffiths. *The Finite Difference Method in Partial Differential Equations.* Wiley, Chichester, 1980.

[169] D. P. Mitchell. Spectrally optimal sampling for distribution ray tracing. *SIGGRAPH Compututer Graphics*, 25(4):157–164, July 1991.

[170] T. Mitsa and K. J. Parker. Digital halftoning using a blue noise mask. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 2809–2812, Apr. 1991.

[171] K. T. Mullen. The contrast sensitivity of human colour vision to red-green and blue-yellow chromatic gratings. *Journal of Physiology*, 359:381–400, 1985.

[172] K. Myszkowski. The visible differences predictor: applications to global illumination problems. In G. Drettakis and N. Max, editors, *Rendering Techniques '98*, pages 223–236, Vienna, 1998.

[173] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:565–593, 1986.

[174] R. Näsänen. Visibility of halftone dot textures. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(6):920–924, 1984.

[175] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, Philadelphia, 1992.

[176] M. Nielsen, L. Florack, and R. Deriche. Regularization, scale-space and edge detection filters. *Journal of Mathematical Imaging and Vision*, 7:291–307, 1997.

[177] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2006.

[178] P. Ochs, Y. Chen, T. Brox, and T. Pock. iPiano: Inertial proximal algorithm for nonconvex optimization. *SIAM Journal on Imaging Sciences*, 7:1388–1419, 2014.

[179] V. Ostromoukhov. A simple and efficient error-diffusion algorithm. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 567–572, New York, 2001.

[180] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[181] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, Mar. 1982.

[182] W.-M. Pang, Y. Qu, T.-T. Wong, D. Cohen-Or, and P.-A. Heng. Structure-aware halftoning. *ACM Transactions on Graphics*, 27(3):1—8, Aug. 2008.

[183] T. N. Pappas and D. L. Neuhoff. Least-squares model-based halftoning. *IEEE Transactions on Image Processing*, 8(8):1102–1116, Aug. 1999.

[184] E. Peli, J. Yang, and R. B. Goldstein. Image invariance with changes in size: the role of peripheral contrast thresholds. *Journal of the Optical Society of America*, 8 (11):1762–1774, Nov. 1991.

[185] W. B. Pennebaker and J. L. Mitchell. *JPEG: Still Image Data Compression Standard*. Springer, New York, 1992.

[186] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[187] P. Peter. Fast inpainting-based compression: Combining Shepard interpolation with joint inpainting and prediction. In *Proc. 2019 IEEE International Conference on Image Processing*, pages 3557–3561, Taipei, Taiwan, 2019.

[188] P. Peter. A Wasserstein GAN for joint learning of inpainting and its spatial optimisation. In H. Wang, W. Lin, P. Manoranjan, G. Xiao, K. Chan, X. Wang, G. Ping, and H. Jiang, editors, *Image and Video Technology*, volume 13763 of *Lecture Notes in Computer Science*, pages 132–145. Springer, Cham, 2023.

[189] P. Peter and J. Weickert. Compressing images with diffusion- and exemplar-based inpainting. In J. Aujol, M. Nikolova, and N. Papadakis, editors, *Scale Space and Variational Methods in Computer Vision*, volume 9087 of *Lecture Notes in Computer Science*, pages 154–165. Springer, Berlin, 2015.

[190] P. Peter, C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert. Beyond pure quality: Progressive mode, region of interest coding and real time video decoding in PDE-based image compression. *Journal of Visual Communication and Image Representation*, 31:256–265, Aug. 2015.

[191] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert. Evaluating the true potential of diffusion-based inpainting in a compression context. *Signal Processing: Image Communication*, 46:40–53, Aug. 2016.

[192] P. Peter, K. Schrader, T. Alt, and J. Weickert. Deep spatial and tonal data optimisation for homogeneous diffusion inpainting. *Pattern Analysis and Applications*, 26(4):1585–1600, Nov. 2023.

[193] M. Pharr, W. Jakob, and G. Humphreys. *Physically Based Rendering: From Theory To Implementation.* Morgan Kaufmann Publishers Inc., third edition, 2016.

[194] G. Plonka, S. Hoffmann, and J. Weickert. Pseudo-inverses of difference matrices and their application to sparse signal approximation. *Linear Algebra and its Applications*, 503:26–47, Aug. 2016.

[195] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Texts and Monographs in Computer Science. Springer, New York, 1985.

[196] T. Preußer and M. Rumpf. An adaptive finite element method for large scale image processing. In M. Nielsen, P. Johansen, O. F. Olsen, and J. Weickert, editors, *Scale-Space Theories in Computer Vision*, volume 1682 of *Lecture Notes in Computer Science*, pages 223–234. Springer, Berlin, 1999.

[197] M. Ramasubramanian, S. N. Pattanaik, and D. P. Greenberg. A perceptually based physical error metric for realistic image synthesis. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 73–82, USA, 1999.

[198] K. R. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications.* Academic Press, San Diego, 1990.

[199] A. Rashid, G. Christine, and M. Nasir. 5.5 - Lossy image compression: JPEG and JPEG2000 standards. In A. Bovik, editor, *Handbook of Image and Video Processing*, Communications, Networking and Multimedia, pages 709–XXII. Academic Press, Burlington, second edition, 2005.

[200] G. Rath and A. Sahoo. A comparative study of some greedy pursuit algorithms for sparse approximation. In *2009 17th European Signal Processing Conference*, pages 398–402, 2009.

[201] M. Roberts. The unreasonable effectiveness of quasirandom sequences, Apr. 2018. URL `http://extremelearning.com.au/unreasonable-effectiveness-of-quasirandom-sequences/`.

[202] J. Robson and N. Graham. Probability summation and regional variation in contrast sensitivity across the visual field. *Vision Research*, 21(3):409–418, 1981.

[203] Y. Romano, M. Elad, and P. Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

[204] G. Rong and T.-S. Tan. Jump flooding in GPU with applications to Voronoi diagram and distance transform. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, I3D '06, pages 109–116, New York, 2006.

[205] L. Ruthotto and E. Haber. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62:352–364, Apr. 2020.

[206] Y. Saad. *Iterative Methods for Sparse Linear Systems.* SIAM, Philadelphia, second edition, 2003.

[207] O. Scherzer and J. Weickert. Relations between regularization and diffusion filtering. *Journal of Mathematical Imaging and Vision*, 12(1):43–63, Feb. 2000.

[208] M. I. Schlesinger and V. Hlaváč. *Ten Lectures on Statistical and Structural Pattern Recognition*, volume 24 of *Computational Imaging and Vision.* Kluwer, Dordrecht, 2002.

[209] C. Schmaltz, P. Peter, M. Mainberger, F. Ebel, J. Weickert, and A. Bruhn. Understanding, optimising, and extending data compression with anisotropic diffusion. *International Journal of Computer Vision*, 108(3):222–240, July 2014.

[210] K. Schnass and P. Vandergheynst. Dictionary preconditioning for greedy algorithms. *IEEE Transactions on Signal Processing*, 56(5):1994–2002, 2008.

[211] M. Schneider, P. Peter, S. Hoffmann, J. Weickert, and E. Meinhardt-Llopis. Gradients versus grey values for sparse image reconstruction and inpainting-based compression. In J. Blanc-Talon, C. Distante, W. Philips, D. Popescu, and P. Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 10016 of *Lecture Notes in Computer Science*, pages 1–13. Springer, Cham, 2016.

[212] C.-B. Schönlieb. *Partial Differential Equation Methods for Image Inpainting.* Cambridge University Press, Cambridge, UK, 2015.

[213] K. Schrader, P. Peter, N. Kämper, and J. Weickert. Efficient neural generation of 4K masks for homogeneous diffusion inpainting. In L. Calatroni, M. Donatelli, S. Morigi, M. Prato, and M. Santavesaria, editors, *Scale Space and Variational Methods in Computer Vision*, volume 14009 of *Lecture Notes in Computer Science*, pages 16–28. Springer, Cham, 2023.

[214] T. Schütze and H. Schwetlick. Bivariate free knot splines. *BIT Numerical Mathematics*, 43(9):153–178, Mar. 2003.

[215] G. Singh, C. Öztireli, A. G. Ahmed, D. Coeurjolly, K. Subr, O. Deussen, V. Ostromoukhov, R. Ramamoorthi, and W. Jarosz. Analysis of sample correlations for Monte Carlo rendering. *Computer Graphics Forum*, 38(2):473–491, 2019.

[216] A. Solé, V. Caselles, G. Sapiro, and F. Arandiga. Morse description and geometric encoding of digital elevation maps. *IEEE Transactions on Image Processing*, 13 (9):1245–1262, Sept. 2004.

[217] G. d. S. Souza, B. D. Gomes, and L. C. L. Silveira. Comparative neurophysiology of spatial luminance contrast sensitivity. *Psychology & Neuroscience*, 4:29–48, June 2011.

[218] K. E. Spaulding, R. L. Miller, and J. S. Schildkraut. Methods for generating blue-noise dither matrices for digital halftoning. *Journal of Electronic Imaging*, 6 (2):208–230, 1997.

[219] J. Sporring, M. Nielsen, L. Florack, and P. Johansen, editors. *Gaussian Scale-Space Theory*, volume 8 of *Computational Imaging and Vision*. Kluwer, Dordrecht, 1997.

[220] A. St-Cyr, M. J. Gander, and S. J. Thomas. Optimized multiplicative, additive, and restricted additive Schwarz preconditioning. *SIAM Journal on Scientific Computing*, 29(6):2402–2425, 2007.

[221] T. Steihaug. The Conjugate Gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983.

[222] 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*, 22(12):1649–1668, 2012.

[223] J. R. Sullivan, L. A. Ray, and R. Miller. Design of minimum visual modulation halftone patterns. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(1): 33–38, Jan. 1991.

[224] X.-C. Tai, J. Hahn, and G. J. Chung. A fast algorithm for Euler's elastica model using augmented Lagrangian method. *SIAM Journal on Imaging Sciences*, 4(1): 313–344, 2011.

[225] D. S. Taubman and M. W. Marcellin. *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer, Boston, 2002.

[226] A. Theljani, Z. Belhachmi, M. Kallel, and M. Moakher. A multiscale fourth-order model for the image inpainting and low-dimensional sets recovery. *Mathematical Methods in the Applied Sciences*, 40(10):3637–3650, Dec. 2016.

[227] A. Toselli and O. Widlund. *Domain Decomposition Methods - Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2005.

[228] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid.* Academic Press, San Diego, 2001.

[229] D. Tschumperlé, C. Porquet, and A. Mahboubi. Reconstruction of smooth 3D color functions from keypoints: Application to lossy compression and exemplar-based generation of color LUTs. *SIAM Journal on Imaging Sciences*, 13(3):1511–1535, 2020.

[230] R. S. Tuminaro, H. F. Walker, and J. N. Shadid. On backtracking failure in Newton–GMRES methods with a demonstration for the Navier–Stokes equations. *Journal of Computational Physics*, 180(2):549–558, 2002.

[231] R. A. Ulichney. *Digital Halftoning.* Cambridge, Massachusetts, 1987.

[232] R. A. Ulichney. Void-and-cluster method for dither array generation. In J. P. Allebach and B. E. Rogowitz, editors, *Human Vision, Visual Processing, and Digital Display IV*, volume 1913, pages 332–343. SPIE Press, 1993.

[233] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.

[234] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-Play priors for model based reconstruction. In *Proc. 2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948, Austin, 2013.

[235] V. Volevich, K. Myszkowski, A. Khodulev, and E. A. Kopylov. Using the visual differences predictor to improve performance of progressive global illumination computation. *ACM Transactions on Graphics*, 19(2):122–161, 2000.

[236] J. Weickert. *Anisotropic Diffusion in Image Processing.* Teubner, Stuttgart, 1998.

[237] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, Dec. 2001.

[238] J. Weickert and M. Welk. Tensor field interpolation with PDEs. In J. Weickert and H. Hagen, editors, *Visualization and Processing of Tensor Fields*, pages 315–325. Springer, Berlin, 2006.

[239] J. Weickert, S. Ishikawa, and A. Imiya. Linear scale-space has first been proposed in Japan. *Journal of Mathematical Imaging and Vision*, 10(3):237–252, May 1999.

[240] J. Weickert, G. Steidl, P. Mrázek, M. Welk, and T. Brox. Diffusion filters and wavelets: What can they learn from each other? In N. Paragios, Y. Chen, and O. Faugeras, editors, *Handbook of Mathematical Models in Computer Vision*, pages 3–16. Springer, New York, 2006.

[241] J. Weickert, M. Welk, and M. Wickert. L^2-stable nonstandard finite differences for anisotropic diffusion. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 390–391. Springer, Berlin, 2013.

[242] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn. Cyclic schemes for PDE-based image analysis. *International Journal of Computer Vision*, 118(3):275–299, July 2016.

[243] P. Weinzaepfel, H. Jégou, and P. Pérez. Reconstructing an image from its local descriptors. In *Proc. 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 337–344, Colorado Springs, June 2011.

[244] M. Welk. Diffusion, pre-smoothing and gradient descent. In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon, editors, *Scale Space and Variational Methods in Computer Vision*, volume 12679 of *Lecture Notes in Computer Science*, pages 78–90. Springer, Cham, 2021.

[245] H. Werner. Studies on contour: I. Qualitative analyses. *The American Journal of Psychology*, 47(1):40–64, Jan. 1935.

[246] P. Wesseling. *An Introduction to Multigrid Methods.* R. T. Edwards, Flourtown, 2004.

[247] G. Westheimer. The eye as an optical instrument. In K. Boff, L. Kaufman, and J. Thomas, editors, *Handbook of Perception and Human Performance: 1. Sensory Processes and Perception*, pages 4.1–4.20. Wiley, New York, 1986.

[248] N. Weyrich and G. T. Warhola. Wavelet shrinkage and generalized cross validation for image denoising. *IEEE Transactions on Image Processing*, 7(1):82–90, Aug. 1998.

[249] S. Wuerger, M. Ashraf, M. Kim, J. Martinovic, M. Pérez-Ortiz, and R. K. Mantiuk. Spatio-chromatic contrast sensitivity under mesopic and photopic light levels. *Journal of Vision*, 20(4):23–23, Apr. 2020.

[250] J. Xu, X.-C. Tai, and L.-L. Wang. A two-level domain decomposition method for image restoration. *Inverse Problems and Imaging*, 4(3):523–545, Aug. 2010.

[251] J. Xu, H. Chang, and J. Qin. Domain decomposition method for image deblurring. *Journal of Computational and Applied Mathematics*, 271:401–414, 2014.

[252] Y. Zeevi and D. Rotem. Image reconstruction from zero-crossings. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34:1269–1277, 1986.

[253] M. Zwicker, W. Jarosz, J. Lehtinen, B. Moon, R. Ramamoorthi, F. Rousselle, P. Sen, C. Soler, and S.-E. Yoon. Recent advances in adaptive sampling and reconstruction for Monte Carlo rendering. *Computer Graphics Forum*, 34(2): 667–681, 2015.

Appendix A

Denoising by Inpainting

A.1 Proof of Theorem 5

We derive the expression for the stated probability in Theorem 5 of Chapter 5 for step $k+1$ here. At the beginning of step $k+1$, Algorithm 3 has already inserted k mask pixels yielding the mask $\boldsymbol{c}^k$. At the end of step $k+1$ we want to have inserted a new mask pixel that is not in $\boldsymbol{c}^k$. Consequently, we can select a pixel only from the set $\mathcal{I}^k$ of remaining empty mask pixel locations, with $|\mathcal{I}^k| = N-k$. The algorithm samples a set $\mathcal{X}$ of α distinct candidates from $\mathcal{I}^k$ uniformly at random (there are C_α^{N-k} different ways to do so):

$$\mathcal{X} = \{X_1, \dots, X_\alpha \in \mathcal{I}^k \,:\, X_i \neq X_j \text{ for } i \neq j\}. \tag{A.1}$$

Then one chooses the candidate $X^* \in \mathcal{X}$ with lowest reconstruction error (w.r.t. the noisy image $\boldsymbol{f}$):

$$X^* \in \mathcal{X}^* = \operatorname*{arg\,min}_{X \in \mathcal{X}} E^k(X), \quad E^k(X) := \|\boldsymbol{r}(\boldsymbol{c}^k + \boldsymbol{e}_X, \boldsymbol{f}) - \boldsymbol{f}\|_2^2, \tag{A.2}$$

where $\boldsymbol{e}_X \in \mathbb{R}^N$ is the zero vector modified with a one at the location corresponding to mask point X. The minimiser does not have to be unique; in fact the set of minimisers

$$\mathcal{X}^* = \{X_i \in \mathcal{X} \,:\, E^k(X_i) = \min_{X \in \mathcal{X}} E^k(X)\} \tag{A.3}$$

may have more than one element ($|\mathcal{X}^*| > 1$) in which case we choose X^* uniformly at random from $\mathcal{X}^*$ with probability $\frac{1}{|\mathcal{X}^*|}$. This completes step $k+1$, now with a specific $X^* = x^*$ and corresponding mask $\boldsymbol{c}^{k+1} = \boldsymbol{c}^k + \boldsymbol{e}_{x^*}$. If the desired number of mask points have been achieved the algorithm ends, otherwise one proceeds to step $k+2$ in the exact same manner.

After we have inserted mask pixel $x^* \in \mathcal{I}^k$ we want to be able to compute the probability $\Pr(X^* = x^*)$ of this occurring. This is equal to the probability of x^* having been selected as a candidate:

$$\Pr(x^* \in \mathcal{X}) = C_{\alpha-1}^{N-k-1} / C_\alpha^{N-k} = \frac{\alpha}{N-k}, \tag{A.4}$$

multiplied by the probability $\Pr(x^* \in \mathcal{X}^* \,|\, x^* \in \mathcal{X})$ that x^* ends up in $\mathcal{X}^*$, which is in turn multiplied by the probability $\Pr(X^* = x^* \,|\, x^* \in \mathcal{X}^*) = \frac{1}{|\mathcal{X}^*|}$ of having picked x^* from

$\mathcal{X}^*$ uniformly at random. We thus have the following chain of conditional probabilities:

$$\begin{aligned}\Pr(X^* = x^*) &= \Pr(X^* = x^* \,|\, x^* \in \mathcal{X}^*) \Pr(x^* \in \mathcal{X}^*) \\ &= \Pr(X^* = x^* \,|\, x^* \in \mathcal{X}^*) \Pr(x^* \in \mathcal{X}^* \,|\, x^* \in \mathcal{X}) \Pr(x^* \in \mathcal{X}) \\ &= \frac{1}{|\mathcal{X}^*|} \Pr(x^* \in \mathcal{X}^* \,|\, x^* \in \mathcal{X}) \frac{\alpha}{N-k}.\end{aligned} \tag{A.5}$$

We can write the terms involving $\mathcal{X}^*$ in the following manner:

$$\frac{1}{|\mathcal{X}^*|} \Pr(x^* \in \mathcal{X}^* \,|\, x^* \in \mathcal{X}) = \sum_{\beta=1}^{\alpha} \frac{1}{\beta} \Pr(x^* \in \mathcal{X}^* \wedge |\mathcal{X}^*| = \beta \,|\, x^* \in \mathcal{X}). \tag{A.6}$$

The probability on the right-hand side can be rewritten as requiring β of the candidates to have energy equal to $E^k(x^*)$ and the remaining $\alpha - \beta$ having a strictly larger energy:

$$\begin{gathered}\Pr(x^* \in \mathcal{X}^* \wedge |\mathcal{X}^*| = \beta \,|\, x^* \in \mathcal{X}) = \\ = \Pr\left(\left(\bigwedge_{i=1}^{\beta} E^k(X_i) = E^k(x^*)\right) \wedge \left(\bigwedge_{j=\beta+1}^{\alpha} E^k(X_j) > E^k(x^*)\right) \middle|\, x^* \in \mathcal{X}\right).\end{gathered} \tag{A.7}$$

To compute the above probabilities we would need to know the total number of pixels from $\mathcal{I}^k$ with energy equal to $E^k(x^*)$:

$$N_{eq} \coloneqq |\{x \in \mathcal{I}^k \,:\, E^k(x) = E^k(x^*)\}|, \tag{A.8}$$

and the total number of pixels from $\mathcal{I}^k$ having a strictly higher energy:

$$N_{gt} \coloneqq |\{x \in \mathcal{I}^k \,:\, E^k(x) > E^k(x^*)\}|. \tag{A.9}$$

From the requirement $|\mathcal{X}^*| = \beta$, it follows that we need to choose β pixels that have energy equal to $E^k(x^*)$. However, $x^* \in \mathcal{X}$ so $E^k(X) = E^k(x^*)$ with probability 1 for at least one candidate $X = x^*$. Then $\beta - 1$ elements X_i remain to be selected from $N_{eq} - 1$ locations, the total number of possibilities being $C_{\beta-1}^{N_{eq}-1}$. Finally the remaining $\alpha - \beta$ candidates must be selected from N_{gt} locations, resulting in $C_{\alpha-\beta}^{N_{gt}}$ options. Using this we can compute the probability

$$\frac{1}{|\mathcal{X}^*|} \Pr(x^* \in \mathcal{X}^* \,|\, x^* \in \mathcal{X}) = \sum_{\beta=1}^{\alpha} \frac{1}{\beta} \frac{C_{\beta-1}^{N_{eq}-1} C_{\alpha-\beta}^{N_{gt}}}{C_{\alpha-1}^{N-k-1}}. \tag{A.10}$$

Ultimately we get the following probability for step $k+1$:

$$\Pr(X^* = x^*) = \frac{\alpha}{N-k} \sum_{\beta=1}^{\alpha} \frac{1}{\beta} \frac{C_{\beta-1}^{N_{eq}-1} C_{\alpha-\beta}^{N_{gt}}}{C_{\alpha-1}^{N-k-1}} = \sum_{\beta=1}^{\alpha} \frac{1}{\beta} \frac{C_{\beta-1}^{N_{eq}-1} C_{\alpha-\beta}^{N_{gt}}}{C_{\alpha}^{N-k}}. \tag{A.11}$$

Through the probabilistic densification procedure the exact same mask $\boldsymbol{c}$, with $\|\boldsymbol{c}\|_0$ mask pixels, can be constructed in $\|\boldsymbol{c}\|_0!$ different ways (the same set of mask pixels being introduced in all possible orders). That is, we get the probability mass function $p_\sigma(\boldsymbol{c}|\boldsymbol{f})$ over masks that also retain the order of insertion of their mask pixels (e.g., we can modify $\boldsymbol{c}$ by setting entries equal to one, to be equal to k: the step in which those were inserted). To get the usual probability mass function over binary masks we need to sum up the above probabilities over all $\|\boldsymbol{c}\|_0!$ permutations of point insertion orders. The main issue for practicality is that N_{eq} and N_{gt} must be known, which would require evaluating all possible $|\mathcal{I}^k| = N - k$ inpaintings for a single step. Nevertheless, Monte Carlo can be used to estimate the probabilities.

A.2 Probability for Error Diffusion Masks

Error diffusion halftoning (e.g., Floyd-Steinberg dithering [85]) can be used to produce a binary mask $\boldsymbol{c} \in \{0,1\}^N$ from a continuous density image $\boldsymbol{d} \in [0,1]^N$. The process involves iterating over the image pixels (e.g., in serpentine order), binarising a single pixel at a given step, and then diffusing the error arising from the binarisation to the set of currently non-visited pixels. This results in a sequence of images $\boldsymbol{d} = \boldsymbol{d}^1, \boldsymbol{d}^2, \ldots, \boldsymbol{d}^{N+1} = \boldsymbol{c}$. The binarisation happens according to a thresholding step, which usually reads:

$$c_k = d_k^{k+1} = \begin{cases} 0 & \text{for } d_k^k < 0.5, \\ 1 & \text{for } d_k^k \geq 0.5. \end{cases} \tag{A.12}$$

Since we want to get multiple masks stochastically, we randomise the process by sampling a uniform random number $v_k \in [0,1]$ for pixel k, and then perform thresholding:

$$c_k = d_k^{k+1} = \begin{cases} 0 & \text{for } d_k^k < v_k, \\ 1 & \text{for } d_k^k \geq v_k. \end{cases} \tag{A.13}$$

Then the probability mass function for mask $\boldsymbol{c}$ constructed from density image $\boldsymbol{d}$ is

$$p_{\boldsymbol{d}}(\boldsymbol{c}) = \frac{1}{P} \prod_{k=1}^{N} (d_k^k(\boldsymbol{c}))^{c_k} (1 - d_k^k(\boldsymbol{c}))^{1-c_k}, \quad P = \sum_{\boldsymbol{c} \in \{0,1\}^N} \prod_{k=1}^{N} (d_k^k(\boldsymbol{c}))^{c_k} (1 - d_k^k(\boldsymbol{c}))^{1-c_k}. \tag{A.14}$$

In the above $d_k^k(\boldsymbol{c})$ are assumed to be clamped to $[0,1]$. Note that while this bears similarity to Poisson sampling (see (5.64)), the probability $d_k^k(\boldsymbol{c})$ is conditioned on the probabilities in the k previous steps. Algorithmically it is trivial to compute the numerator of the probability during the error diffusion process.