Piecewise Planar Image Restoration via Gradient Graph Laplacian Regularizer

Yeganeh Gharedaghi

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

JUNE 2024

Abstract

Images in various practical applications often exhibit Piecewise Planar (PWP) characteristics. Common issues such as noise, blur, and distortions arise from sensor limitations, transmission errors, and environmental factors like lighting and sensor quality. Despite advancements in image restoration, significant challenges remain under conditions of poor lighting, camera movement, and occlusions, making PWP image restoration a critical research area. This thesis investigates the restoration of PWP images, focusing on efficiently overcoming these challenges and enhancing image quality in two real-world applications: (i) depth image denoising and (ii) low-light image contrast enhancement. We formulate quadratic objectives regularized by Graph Laplacian Regularizer and Gradient Graph Laplacian Regularizer, tailored for these two applications. These objectives can be efficiently solved in linear time using a conjugate gradient solver and the alternating direction method of multipliers. Experimental results demonstrate that our algorithm achieves competitive image quality while significantly reducing computational complexity.

Contents

List of Figures

Chapter 1

Introduction

Many images in practical applications can be suitably modelled as *Piecewise Planar* (PWP). PWP signals refer to the generalization of piecewise linear signals in one dimension to two dimensions. Formally, a PWP signal consists of a concatenation of multiple signal components, where the domain of all these components collectively partitions the entire domain of the signal. Each signal component has zero second-order derivatives (with respect to x- and y-axis), and thus each component is planar, making the entire signal piecewise planar. Examples of PWP images include depth images [3, 4] and the illumination component of a natural image in a Retinex decomposition [5] (to be discussed). Acquired images are often degraded by various types of noise, blur, distortion, or sampling, due to limitations in the sensing device and/or transmission errors. The performance of imaging sensors is affected by various environmental factors, such as lighting conditions during image acquisition and the quality of the sensing technology. Despite many advances in the field of image restoration, situations of poor illumination, camera motion, or occlusion are still challenging for most restoration methods, making it difficult to obtain good results. Further, image restoration techniques employing priors that promote *piecewise*

constant (PWC) signal reconstruction such as *total variation* (TV) [6, 7] exhibit an undesirable "staircase" effect when used on PWP images—artificially created edges between adjacent constant pieces that do not exist in the original signals.

In scenarios where pixel intensities are gradually changing, an ideal prior should promote PWP signal reconstruction. In this thesis, we focus on restoring signals with PWP characteristics for two targeted image processing applications: i) depth image denoising, and ii) joint image denoising / contrast enhancement.

1.1 Depth Image Denoising

Depth images record the per-pixel Euclidean distances between the image sensor and the physical objects in the 3D scene. They are often considered PWP because no textural information is captured, and physical surfaces of objects tend to be well approximated as flat planes [3, 4]. As shown in Fig. 1.1, the changes in disparity map—representing pixel differences between corresponding points in stereo images and thus the inverse of depth estimates—are often gradual with occasional discontinuities.

Using a captured depth image, a *Point Cloud* can be synthesized by back-projecting each pixel into the 3D space. However, these depth images frequently are noise-corrupted, which would significantly degrade the subsequent constructed point clouds. Such noise not only distorts visual quality but also adversely impacts the performance of point-cloud-based applications, such as autonomous driving [8] and virtual / augmented reality (AR/VR) [9]. Addressing these challenges in depth image denoising is

pivotal for improving the accuracy and reliability of 3D scene interpretations. Treating depth images with PWP priors is particularly beneficial, as it simplifies the complex geometries of real-world scenes, thereby facilitating a more efficient and accurate reduction of noise.

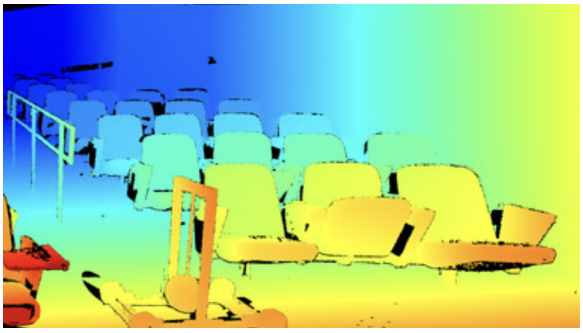


Figure 1.1: Disparity map of `Pendulum` selected from Middlebury 2021 Mobile stereo dataset [1], showing the gradual changes in depth estimations of indoor scenes.

1.2 Low-light Image Contrast Enhancement

In low light conditions, natural image sensors face a significant issue due to the reduced number of photons captured per pixel, leading to noticeable acquisition noise. This issue becomes particularly pronounced when applying contrast enhancement algorithms [10]. While these algorithms are designed to selectively brighten areas of an image to enhance visual appeal, they inadvertently amplify this inherent noise. Consequently, this can lead to a degradation in the overall image quality. Thus, addressing image denoising and contrast enhancement together yields better results, especially under low-light conditions.

1.3 Related Works in Image Restoration

Image restoration techniques can be broadly categorized into model-based and data-driven approaches. Even though data-driven approaches that use neural networks have recently demonstrated superior performance [11–15], they require substantial amounts of training data and computational resources, and can sometimes be seen as "black boxes" with little interpretability. On the other hand, model-based approaches rely on well-defined mathematical models and are more interpretable. Examples of such methods include TV [6,7], sparse coding [16,17], or *graph Laplacian regularizers* (GLR) [18,19]—a graph smoothness prior from a recent *graph signal processing* (GSP) field that studies mathematical tools for spectral processing and analysis of discrete signals on irregular kernels described by graphs [20,21]. TV and GLR promote PWC signal reconstruction and are not directly suitable for PWP signals—direct application can result in a staircase effect. A generalization of TV called *total generalized variation* (TGV) [22] can promote PWP signal reconstruction, but the non-differentiable ℓ_1 norm means more complicated and time-consuming optimization algorithms, such as *proximal gradient* (PG) [23]. Recently, a generalization of GLR called *gradient graph Laplacian regulirizer* (GGLR) [24] was proposed and demonstrated PWP signal reconstruction, while retaining the convenient ℓ_2-norm that is amenable to fast optimization. In this thesis, our focus is on tailoring GGLR to restore PWP images for two application scenarios: i) depth image denoising, and ii) joint image denoising / contrast enhancement.

The thesis is structured as follows. Chapter 2 introduces the fundamental concepts and notations in GSP, setting the foundation for this thesis. Chapter 3 provides an in-depth review of related methodologies such as TV, TGV, and sparse coding. These

methods, significant in image restoration, are examined particularly in the context of PWP signals, the focus of our study. The final section of Chapter 3 engages in a critical analysis of the challenges associated with GGLR optimization, with a special emphasis on enhancing efficiency. Chapter 4 details the proposed algorithm, beginning with a discussion of the algorithm for PWP signals and continuous PWP signal reconstruction on a one-dimensional signal. Subsequently, we adapt this approach for images. Finally, Chapter 5 presents the outcomes of our algorithms in depth image denoising and low-light image contrast enhancement, comparing them with existing competing methods.

Chapter 2

Preliminaries

GSP extends spectral analysis tools in conventional *discrete signal processing* (DSP) like transforms and wavelets to the graph domain [20, 21]. GSP tools have been successfully applied to a wide range of practical applications, including image compression, denoising, deblurring, dequantization, and point cloud denoising [25–29]. A graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathbf{W})$ is composed of N nodes $\mathcal{N} = \{1, \ldots, N\}$ and edges $\mathcal{E}$ connecting them, where edge $(i, j) \in \mathcal{E}$ has weight $w_{i,j} = W_{i,j}$. Assuming that the edges are undirected, the adjacency matrix $\mathbf{W}$ is symmetric. The *combinatorial graph Laplacian matrix* $\mathbf{L}$ is defined as $\mathbf{L} \triangleq \text{diag}(\mathbf{W1}) - \mathbf{W}$, where $\mathbf{1}$ is an all-one vector of appropriate length, and $\text{diag}(\mathbf{W1})$ denotes a diagonal matrix with vector $\mathbf{W1}$ as diagonal terms. If the graph contains self-loops, i.e., $w_{i,i} \neq 0$ for some i, then the *generalized graph Laplacian matrix* $\mathbf{L}_g$, defined as $\mathbf{L}_g \triangleq \text{diag}(\mathbf{W1}) - \mathbf{W} + \text{diag}(\mathbf{W})$, is commonly used. In both cases, graph Laplacian is provably *positive semi-definite* (PSD)—*i.e.*, $\mathbf{x}^\top \mathbf{L} \mathbf{x} \geq 0, \forall \mathbf{x}$—if the edges are non-negative [21]. Eigen-decomposing $\mathbf{L} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$, where $\mathbf{V} = [\mathbf{v}_1; \cdots; \mathbf{v}_N]$ and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \cdots, \lambda_N)$, leads to eigen-pairs $(\lambda_k, \mathbf{v}_k)$ for $k \in \{1, \ldots, N\}$, interpreted as the k-th *graph frequency* and *Fourier mode* for graph $\mathcal{G}$ [20].

2.1 Graph Laplacian Regularizer

$\mathbf{x}^\top \mathbf{L}\mathbf{x}$ is called the GLR [18] and is used for regularization in different graph signal restoration problems, including image denoising [18], JPEG dequantization [30], point cloud denoising [29, 31], and super-resolution [32]. GLR can be similarly defined for graphs with self-loop using the generalized graph Laplacian $\mathbf{L}_g$. GLR can be interpreted in both nodal and spectral domains:

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \sum_{(i,j)\in\mathcal{E}} w_{i,j}(x_i - x_j)^2 = \sum_{k=1}^{N} \lambda_k \tilde{x}_k^2. \tag{2.1}$$

Signal energy $\tilde{\mathbf{x}}$ represents the graph signal $\mathbf{x}$ transformed into the spectral domain and is calculated as $\tilde{\mathbf{x}} = \mathbf{V}^\top \mathbf{x}$. In the spectral domain, minimizing $\mathbf{x}^\top \mathbf{L}\mathbf{x}$ translates to minimizing $\sum_{k=1}^{N} \lambda_k \tilde{x}_k^2$, which biases the signal energies $\tilde{x}_k^2$ to the low frequencies λ_k—small eigenvalues of $\mathbf{L}$, where the 0 frequency for a positive connected graph $\mathcal{G}$ without self-loops is the constant signal. Thus, signal $\mathbf{x}$ should be roughly low-pass, where energies $\tilde{x}_k^2$ for large k are mostly zeros. On the other hand, in the nodal domain, minimizing $\mathbf{x}^\top \mathbf{L}\mathbf{x}$ translates to minimizing $\sum_{(i,j)\in\mathcal{E}} w_{i,j}(x_i - x_j)^2$, which forces neighbouring samples (x_i, x_j) to be similar for large edge weights $w_{i,j}$.

In the *signal-dependent* variant of GLR, called SDGLR, each edge weight $w_{i,j}$ is defined as

$$w_{i,j} \triangleq \exp\left(-\frac{(x_i - x_j)^2}{\sigma^2}\right), \tag{2.2}$$

where $\sigma > 0$ is a user-defined parameter. A term $w_{i,j}(x_i - x_j)^2$ in the sum (2.1) is minimized when: (i) $|x_i - x_j| \to 0$, or (ii) $|x_i - x_j| \to \infty$, in which case $w_{i,j} \to 0$. This results in convergence to a *piecewise constant* (PWC) reconstructed signal [18, 30].

2.2 Gradient Graph Laplacian Regularizer

A recent variant of GLR is GGLR [33] and has been shown to promote *piecewise planar* (PWP) signal reconstruction, *e.g.*, an image patch with PWC vertical and horizontal gradients. Considering a row (column) of N pixels $\mathbf{x} \in \mathbb{R}^N$, *gradient operator* $\mathbf{F} \in \mathbb{R}^{(N-1)\times N}$ is defined as

$$F_{i,j} \triangleq \begin{cases} 1 & \text{if } i = j \\ -1 & \text{if } i = j+1 \\ 0 & \text{otherwise} \end{cases} . \tag{2.3}$$

The gradient vector $\boldsymbol{\alpha}$ is calculated using the gradient operator $\mathbf{F}$ by setting $\boldsymbol{\alpha} = \mathbf{F}\mathbf{x} \in \mathbb{R}^{N-1}$. The *gradient graph* $\bar{\mathcal{G}}$ has $N-1$ nodes, where each node represents an element in $\boldsymbol{\alpha}$. In the signal-independent case, adjacency matrix $\bar{\mathbf{W}}$ for $\bar{\mathcal{G}}$ has $\bar{w}_{i,i+1} = 1$ for all $1 \leq i < N-1$. We can define the signal-dependent variant of GGLR, called SDGGLR, where each edge weight $\bar{w}_{i,j}$ is

$$\bar{w}_{i,j} \triangleq \exp\left(-\frac{(\alpha_i - \alpha_j)^2}{\sigma_\alpha^2}\right). \tag{2.4}$$

Then, *gradient graph Laplacian* $\bar{\mathbf{L}}$ is defined as $\bar{\mathbf{L}} \triangleq \text{diag}(\bar{\mathbf{W}}\mathbf{1}) - \bar{\mathbf{W}}$. Since $\bar{\mathcal{G}}$ is a graph with positive edge weights, the Laplacian $\bar{\mathbf{L}}$ is provably PSD [21]. Now GGLR

can be defined as GLR for gradient vector $\boldsymbol{\alpha}$:

$$\boldsymbol{\alpha}^\top \bar{\mathbf{L}} \boldsymbol{\alpha} = \mathbf{x}^\top \underbrace{\mathbf{F}^\top \bar{\mathbf{L}} \mathbf{F}}_{\mathcal{L}} \mathbf{x} = \mathbf{x}^\top \mathcal{L} \mathbf{x}, \tag{2.5}$$

where $\mathcal{L} = \mathbf{F}^\top \bar{\mathbf{L}} \mathbf{F}$ is called the *gradient-induced nodal graph (GNG) Laplacian* and is provably PSD. Fig. 2.1 illustrates an example of gradient operator $\mathbf{F}$, gradient graph $\bar{\mathcal{G}}$ and GNG for a row of three pixels.

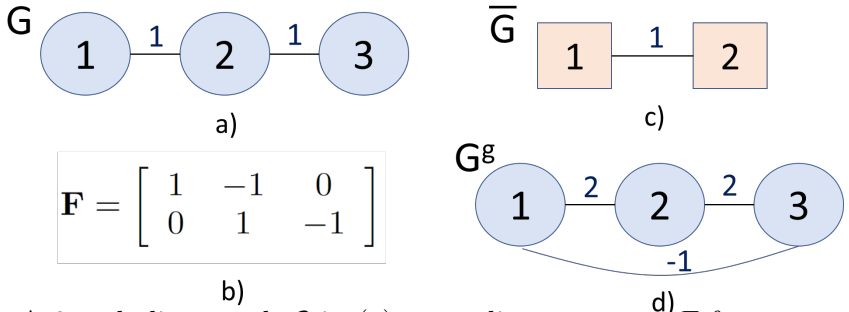


Figure 2.1: A 3-node line graph $\mathcal{G}$ in (a), a gradient operator $\mathcal{F}$ for a row / column of 3 pixels in (b), gradient graph $\bar{\mathcal{G}}$ in (c), and resulting GNG in (d)—a signed graph with positive/negative edges.

Chapter 3

Related Work and Motivation

3.1 Image Restoration

In image restoration problems, a ground truth image $\mathbf{x}$ is corrupted by additive zero-mean Gaussian noise $\mathbf{v}$, and possibly a linear transformation $\mathbf{H}$, resulting in observation $\mathbf{y}$:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}. \tag{3.1}$$

In a denoising scenario, $\mathbf{H}$ is the identity matrix. The image restoration problem is to recover image $\mathbf{x}$ given only observation $\mathbf{y}$.

First, in Section 3.1.1, we discuss some early methods in image denoising that did not incorporate specific priors or assumptions on PWP image characteristics. These methods often fell short of effectively reconstructing images with PWP characteristics and resulted in overly smoothed outcomes.

3.1.1 Spatial Domain Filtering

Spatial domain filters are classic restoration techniques for denoising. They can be categorized into linear and non-linear filters. Among linear filters, mean filter and Gaussian filter are notable [34]. A mean filter sets the restored image's value at each pixel to the arithmetic mean of pixels within a neighbourhood. In contrast, a Gaussian filter assigns kernel weights based on an approximation to the probability mass of a Gaussian centred at the target pixel. Spatial linear filters often struggle to preserve image details like edges and corners that are crucial in PWP image reconstruction.

Non-linear filters, including median filter [34] and bilateral filter (BF) [35], can better preserve image details. A median filter replaces the value of the center pixel with the median of the intensity values in the neighbourhood of that pixel. This idea, while simple, resolves the over-smoothing issue in linear filters. However, it is not targeted for PWP signal reconstruction. A BF, like the mean filter, uses weighted averaging, but the weight coefficients are chosen in a *signal-dependent manner*. Specifically, denote by $\mathbf{x}$ the restored image, by $\mathbf{y}$ the corrupted image, by $\mathbf{l}_i$ the location of pixel i on the 2D image grid, and by Ω_i a local neighbourhood centred at pixel i. Then BF can be formulated as follows:

$$x_i = \frac{1}{W_p} \sum_{j \in \Omega_i} y_j \cdot f(|y_i - y_j|) \cdot g(||\mathbf{l}_i - \mathbf{l}_j||) \tag{3.2}$$

where W_p is a normalizing factor, and $\|\cdot\|$ denotes a chosen norm. f and g are chosen non-increasing functions with respect to their scalar arguments: the former is a function of differences in pixel intensities, and the latter is a function of spatial distances. Compared to linear filters, these functions are chosen to preserve edges in

the image while reducing noise, thus preventing over-smoothing. When BF is applied iteratively, it may cause an undesirable "staircase" effect in the restored images—PWC signal reconstruction that differs from the original signal characteristics, such as PWP images [24].

3.1.2 Variational Denoising

In variational approaches, filtering is achieved by solving an optimization problem with a well-defined objective. For example, for additive white Gaussian noise (AWGN), the objective is written as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{x}\|_2^2 + \mu\, R(\mathbf{x}), \tag{3.3}$$

where $\|\mathbf{y} - \mathbf{x}\|_2^2$ is the fidelity term and $R(\cdot)$ is a chosen prior for regularization. The parameter μ provides a tradeoff between the fidelity term and the image prior. Variational approaches can be suitable for PWP signal reconstruction if an appropriate prior $R(\cdot)$ can be designed to promote planarity—$R(\mathbf{x})$ computes to a small value if $\mathbf{x}$ is PWP.

3.1.2.1 Sparse-representation

An example of variational denoising methods is sparse coding [17], where an image (or image patch) $\mathbf{x}$ is represented as a linear combination $\mathbf{D}\boldsymbol{\alpha}$ of atoms from an overcomplete dictionary $\mathbf{D}$, and the code vector $\boldsymbol{\alpha}$ is assumed to be sparse. The optimization thus becomes

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \|\mathbf{y} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|. \tag{3.4}$$

This overcomplete dictionary $\mathbf{D}$ can be trained offline from data, or built from known transforms such as the *discrete cosine transform* (DCT). To promote sparsity in $\boldsymbol{\alpha}$, a non-convex ℓ_0-norm can be used, but the problem becomes NP-hard. In such cases, greedy algorithms such as matching pursuit or orthonormal matching pursuit [36] can be used.

An alternative is to convexify ℓ_0-norm to convex ℓ_1-norm, in which case the problem can be solved using basis pursuit [37], *iterative shrinkage-thresholding algorithm* (ISTA) [22], or *fast iterative shrinkage-thresholding algorithm* (FISTA) [38]. Nevertheless, the ℓ_1-norm is non-differentiable, thus requiring iterative algorithms like ISTA and FISTA that are still computation-expensive.

Further, while the dictionary $\mathbf{D}$ can be trained on PWP images, the sparse coding framework is not designed specifically for PWP signal reconstruction. This comes at the expense of added complexity in finding a sparse code vector $\boldsymbol{\alpha}$. In summary, while sparse coding is a general representation framework that is intuitive and powerful, two primary challenges remain: i) the task of acquiring an overcomplete dictionary, essential for these methods, can be quite demanding and complex; and ii) solving the sparse optimization objective, as formulated in (3.4), poses significant difficulties in terms of computational efficiency.

3.1.2.2 Total Variation

Another popular prior in variational denoising is *total variation* (TV)-based regularization [6, 39], which borrows the concept of natural images being locally smooth,

implying that pixel intensities change gradually in most areas. Formally, the (discrete) total variation $\mathrm{R_{TV}}(\mathbf{x})$ is expressed as

$$\mathrm{R_{TV}}(X) = \sum_{j \in \Omega_i} \|x_i - x_j\|_q, \tag{3.5}$$

where Ω_i is the set of four horizontal and vertical adjacent pixels of x_i, and $||\cdot||_q$ is the ℓ_q-norm. A shorthand notation of $\mathrm{R_{TV}}(\mathbf{x})$ is $||\nabla \mathbf{x}||_q$, where $\nabla \mathbf{x}$ is the gradient of $\mathbf{x}$ and is computed by taking the difference between two consecutive samples; for an image, two pixels are considered consecutive samples if their corresponding pixels are either horizontally or vertically adjacent, making it of dimension $2N(N-1)$.

In most cases, ℓ_1-norm is used to induce sparsity in the optimal solution. In this form, $\mathrm{R_{TV}}$ can be solved using ISTA or FISTA. However, similar to the challenges discussed in sparse representations, solving TV remains computationally demanding, primarily due to the non-differentiability of its ℓ_1-norm term, which has no closed-form solutions [40]. Further, the ℓ_1 norm induces a sparse gradient that promotes PWC signals that exhibit an undesirable staircase effect in the denoised images.

In this context, *total generalized variation* (TGV) [41] was introduced as an extension of TV that aims to overcome the staircase effect by considering higher-order derivatives in addition to the first derivative. However, TGV retains the non-differentiable ℓ_1-norm, which is hard to solve efficiently.

3.1.2.3 Graph-based priors

As explained in Chapter 2, SDGLR [18] defined as $\mathrm{R_{SDGLR}}(\mathbf{x}) = \mathbf{x}^\top \mathbf{L}(\mathbf{x})\mathbf{x}$ can be used for denoising. Using $\mathrm{R_{SDGLR}}(\mathbf{x})$, which has a convenient quadratic form, in (3.3) leads to a system of linear equations for solution $\mathbf{x}^*$, that, unlike TV or sparse coding,

can be computed efficiently. Specifically, assuming that the coefficient matrix $\mathbf{I} + \mu\mathbf{L}$ is sparse, positive definite and symmetric, it can be solved in roughly linear time using *conjugate gradient* (CG) [42]. While TV and GLR are effective at denoising and maintaining sharp image edges, they have a shortcoming: a PWC approximation of a planar spatial region results in an undesirable staircase reconstruction.

GGLR, a variant of GLR explained in Chapter 2, was also shown to promote PWP signal reconstruction [33], and is effective in reconstructing data characterized by linear / gradual variations without causing staircase effect. Further, when Laplacian $\mathbf{L}$ is fixed (signal-independent), GGLR is convex and differentiable, leading to an efficient and fast optimization process.

Another graph-based smoothness prior is *graph total variation* (GTV) [27]:

$$\text{GTV}(x) = \sum_{i \sim j} w_{i,j} |x_i - x_j|, \tag{3.6}$$

where the sum is over all pairs of connected nodes i and j. Similar to SDGLR, signal-dependent GTV promotes PWC signal reconstruction. SDGTV is a non-differentiable term, which is solved using an iterative *proximal gradient descent* (PGD) method [23]. Notably, it achieves convergence to a PWC signal in fewer iterations relative to SDGLR, while maintaining comparable performance.

3.1.3 Block Matching and 3D Filtering

Block matching and 3D filtering (BM3D) [43] is an image denoising scheme that denoises small, overlapping blocks of the image and then aggregates them to obtain the final denoised image. Specifically, this method performs block matching to find highly correlated blocks, stacks them, and denoises them together as a 3D tensor,

thus effectively utilizing shared information across multiple pieces of the image that are further apart.

In details, given an image of size N-by-N, BM3D defines $(N-N_1+1)\times(N-N_1+1)$ overlapping blocks of size $N_1 \times N_1$, where $X_{r,c}$ denotes a block with its upper-left pixel being in the r-th row and c-th column of the original image. The method first processes all the blocks $X_{r,c}$ using an initial 2D low-pass filter to obtain intermediate denoised blocks (denoted by $\hat{X}_{r,c}$ for all $1 \leq r \leq N - N_1$ and $1 \leq c \leq N - N_1$) and then uses an ℓ_2 distance metric between them $\|\hat{X}_{r,c} - \hat{X}_{r',c'}\|_2^2$. Next, two blocks $X_{r,c}$ and $X_{r',c'}$ are matched if and only if $\|\hat{X}_{r,c}, \hat{X}_{r',c'}\|_2^2 \leq \tau_m$, where τ_m is a prespecified matching threshold.

For increased tractability, BM3D only considers the top N_2 matched blocks (in terms of the ℓ_2 distance metric) to stack them and create the 3D tensor. Then, it processes the 3D tensor using a combination of Wiener [35] and low-pass filters to obtain denoised 3D tensors, and unpacks the tensor to retrieve all the denoised blocks. Finally, each pixel is reconstructed by taking a weighted average of the pixel values in each of the denoised blocks containing that pixel. Thus, BM3D uses information across similar blocks to enhance its performance.

The time complexity of the algorithm is as follows:

$$\mathcal{O}\left(N^2\mathcal{T}_{2D}(N_1, N_1)\right) + \mathcal{O}\left(N^4(N_1^2 + N_2)\right) + \mathcal{O}\left(N^2\mathcal{T}_{3D}(N_1, N_1, N_2)\right), \tag{3.7}$$

where $\mathcal{T}_{2D}(N_1, N_1)$ is the time complexity of the intermediate 2D denoising step and $\mathcal{T}_{3D}(N_1, N_1, N_2)$ is the time complexity of the 3D denoising step on the $N_1 \times N_1 \times N_2$ tensor. It is important to note that the middle term accounts for block matching by considering all possible block pairs, resulting in an N^4 term that can significantly

contribute to the algorithm's inefficiency.

To circumvent this, instead of using all the block pairs, the algorithm chooses reference blocks by performing a sliding window of size N_s, selecting $\lfloor \frac{N}{N_s} \rfloor^2$ reference blocks. For better tractability, BM3D only defines an $N_l \times N_l$ local neighbourhood around each reference block and only matches blocks within this neighbourhood. Thus, instead of having $\mathcal{O}(N^4)$ block pairs, BM3D takes $\mathcal{O}(\frac{N^4}{N_s^2} N_l^2)$ pairs into consideration, enhancing the total time complexity to:

$$\mathcal{O}\left(N^2 \mathcal{T}_{2D}(N_1, N_1)\right) + \mathcal{O}\left(\frac{N^2}{N_s^2} N_l^2 (N_1^2 + N_2)\right) + \mathcal{O}\left(\frac{N^2}{N_s^2} \mathcal{T}_{3D}(N_1, N_1, N_2)\right). \quad (3.8)$$

Despite this, the complexity of BM3D is still super-linear with respect to the image size.

3.1.4 Non-local Graph Based Transform

Non-local Graph-based transform (NLGBT) [44] builds upon graph-based transforms (GBT) [45]. However, to understand NLGBT, it is essential to first cover GBT briefly. At its core, GBT is a sparse representation algorithm that utilizes the eigenvectors of the graph Laplacian as elements of its complete dictionary. Formally, given a signal $\mathbf{y} \in \mathbb{R}^N$ that we aim to reconstruct, GBT first constructs the graph Laplacian $\mathbf{L} \in \mathbb{R}^{N \times N}$ and then performs an eigen-decomposition to obtain a matrix $\mathbf{U} \in \mathbb{R}^{N \times N}$, where each column represents an eigenvector of $\mathbf{L}$. Here, $\mathbf{U}$ can be considered the complete dictionary (denoted as $\mathbf{D}$ in (3.4)). Consequently, GBT aims to minimize the following objective:

$$\min_{\boldsymbol{\alpha}} \|\mathbf{y} - \mathbf{U}\boldsymbol{\alpha}\|_2^2 + \tau \cdot \|\boldsymbol{\alpha}\|_0 \quad (3.9)$$

where $\|\cdot\|_0$ is the ℓ_0 nuclear norm promoting sparsity in code vector $\boldsymbol{\alpha}$. One can relax the ℓ_0 norm by ℓ_1 norm and efficiently solve it using the split Bergman method [46].

For NLGBT, the algorithm processes images in patches and matches similar patches using an approach similar to BM3D [43]. For a given cluster of matched patches $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\}$, the method first computes an average patch $\bar{\mathbf{y}} \coloneqq \frac{1}{K}\sum_{k=1}^{K}\mathbf{y}_k$, then computes the joint Laplacian $\bar{\mathbf{L}}$ based on average patch $\bar{\mathbf{y}}$. Performing eigen-decomposition on $\bar{\mathbf{L}}$ yields the eigenvector matrix $\bar{\mathbf{U}}$, which serves as the complete dictionary. By jointly optimizing all the similar patches, we can obtain the sparse code vectors $\{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$:

$$\min_{\boldsymbol{\alpha}_1 \ldots \boldsymbol{\alpha}_K} \sum_{k=1}^{K} \|\mathbf{y}_k - \bar{\mathbf{U}}\boldsymbol{\alpha}_k\|_2^2 + \tau \sum_{k=1}^{K} \|\boldsymbol{\alpha}_k\|_0. \tag{3.10}$$

$\{\bar{\mathbf{U}}\alpha_1, \ldots, \bar{\mathbf{U}}\alpha_K\}$ are the denoised patches, and all of these patches are aggregated to produce the final image.

3.2 Contrast Enhancement

As discussed in Chapter 1, imaging researchers have interpreted Land's 1977 seminal Retinex theory in human vision [5] to mean that a recorded pixel intensity is the product of illumination and reflectance components [47, 48]. Specifically, a ground-truth N-by-N observed image patch $\mathbf{Y}$ is a point-by-point multiplication of strictly positive illumination and reflectance components, $\mathbf{L}, \mathbf{R} \in \mathbb{R}_+^{N\times N}$, respectively, plus a

noise term $\mathbf{Z}$:

$$\mathbf{Y} = \mathbf{L} \odot \mathbf{R} + \mathbf{Z}, \tag{3.11}$$

where operator $\odot$ denotes point-by-point multiplication. Reflectance $\mathbf{R}$ depends only on the surface properties of physical objects, which tends to be PWC. In contrast, illumination $\mathbf{L}$ is more generally smooth, characterized by gradual, rather than abrupt, variations.

Image restoration schemes based on Retinex theory typically first separate these two components, then restore each one using suitable signal priors, before combining them to reconstruct the desired image [47–53]. However, computation of the illumination and reflectance components can be expensive. For example, [48] first takes the logarithm of $\mathbf{R}$ and $\mathbf{L}$, and then rewrites (3.11) as $\nabla \log(\mathbf{Y}) = \nabla \log(\mathbf{R}) + \nabla \log(\mathbf{L})$ by computing the gradients. Based on the assumption that $\mathbf{L}$ is spatially smooth, $\nabla \log(\mathbf{L})$ is relatively small. On the other hand, $\nabla \log(\mathbf{R})$ corresponds to sharp details such as edges. To preserve only the sharp details, $\nabla \log(\mathbf{R})$ is approximated by $\delta_t(\nabla \log(\mathbf{Y}))$, where δ_t is a threshold function that retains only those gradients of $\nabla \log(\mathbf{Y})$ that exceed a threshold t while setting all others to zero. Finally, $\nabla \log(\mathbf{R}) - \delta_t(\nabla \log(\mathbf{Y}))$ is minimized to promote a PWC reflectance. This is formulated using an ℓ_1-norm term which is minimized by split Bregman iteration [54].

[49] proposes low-light image enhancement via illumination map estimation (LIME), where the following optimization problem is used to recover the illumination component:

$$\min_{\mathbf{L}} \|\hat{\mathbf{L}} - \mathbf{L}\|_F^2 + \mu \|\mathbf{W} \odot \nabla \mathbf{L}\|_1. \tag{3.12}$$

Here, $\nabla\mathbf{L}$ represents the derivative filter applied horizontally and vertically to the illumination map, while $\hat{\mathbf{L}}$ is the initial illumination map estimation. μ is a tradeoff parameter, balancing the two terms, and $\|\cdot\|_F$ is the Frobenius norm. The weight matrix $\mathbf{W}$ is inversely related to the horizontal (vertical) gradient of the initial illumination map. To address an ℓ_1-norm term together with the gradient operation on $\mathbf{L}$, LIME performs the following optimization process: (i) employ Alternating Direction Method (ADM) [55], where each iteration takes $\mathcal{O}(N^2 \log N)$, (ii) approximate $\|\mathbf{W} \odot \nabla\mathbf{L}\|_1$ with a quadratic term leading to an optimization that has a closed form and can be solved via CG. Finally, LIME applies BM3D as a post-processing denoising method. Note that ADM in LIME makes it time-consuming, as multiple iterations, each of complexity $\Theta(N^2 \log N)$, are needed.

Joint enhancement and denoising method (JED) [50] first reconstructs the illumination component by minimizing an optimization problem that contains a TV term, which, like LIME, is approximated by a quadratic term. Having illumination $\mathbf{L}$ and a corrupted image $\mathbf{Y}$, $\mathbf{Y}/\mathbf{L}$ is the elementwise division of the two matrices, and the reflectance is obtained by solving

$$\min_{\mathbf{R}} \|\mathbf{R} - \mathbf{Y}/\mathbf{L}\|_F^2 + \beta\|\mathbf{W} \odot \nabla\mathbf{R}\|_F^2 + \omega\|\nabla\mathbf{R} - \mathbf{G}\|_F^2, \tag{3.13}$$

where $\mathbf{G}$ is the adjusted gradient of $\mathbf{Y}$ and $\mathbf{W}$ is a weight matrix. The adjusted gradient is a scaled gradient designed to amplify reflectance while restraining noise by balancing the gradients (see Equation 5 of [50] for more details). In this approach, $\mathbf{W}$ is set such that $\mathbf{W}_{i,j} = \frac{1}{|\nabla\mathbf{Y}_{i,j}|+\epsilon}$ to promote smooth gradients. β and ω are tradeoff parameters. Finally, the term $\|\nabla\mathbf{R} - \mathbf{G}\|_F^2$ is used to preserve edges by minimizing the distance between the gradient of the reflectance and that of the observed image.

(3.13), being a convex differentiable function, is solved in closed form by differentiating w.r.t. $\mathbf{R}$ and setting the derivative to $\mathbf{0}$.

Low-rank regularized Retinex model (LR3M) [51] is an extension of JED and includes an additional low-rank prior that minimizes the rank of matrices of similar patches of $\mathbf{R}$. However, low-rank minimization problem is NP-hard, and thus the problem is relaxed by replacing the rank constraint with a nuclear norm—the sum of the singular values—and is solved using an iterative singular-value thresholding method [56]. However, this solution is computationally expensive, due to *singular value decomposition* (SVD) that is required in each iteration.

In all of the mentioned methods above, after the estimation of the illumination $\mathbf{L}$ and the reflectance $\mathbf{R}$, the gamma correction operation is applied in order to adjust the illumination [34], and the contrast-enhanced result $\hat{\mathbf{Y}}_{i,j}$ is generated by

$$\hat{\mathbf{Y}}_{i,j} = (\mathbf{L}_{i,j})^{\gamma}\ \mathbf{R}_{i,j}, \tag{3.14}$$

where $0 < \gamma < 1$ is a pre-chosen parameter. The operation $(\mathbf{L}_{i,j})^{\gamma}$ essentially boosts the illumination component—more enhancement when $\mathbf{L}_{i,j}$ is small, and less enhancement when $\mathbf{L}_{i,j}$ is large.

The primary challenge with these methods lies in their reliance on non-differentiable terms such as ℓ_1-norm in (3.12) and low-rank minimization, making the optimization a potentially time-consuming process. To mitigate these challenges, some methods resort to quadratic approximations of the ℓ_1-norm, which are aimed at improving speed at the expense of reduced accuracy. Thus, a problem formulation that is convex and differentiable by definition is preferred.

Alternatively, deep learning based Retinex schemes are also possible [11,12]. However, they require expensive data training for a large number of network parameters with a large memory footprint and thus are not typically suitable for memory-constrained devices like mobile phones. Further, the effectiveness of these approaches is largely dependent on the quality of the datasets used for training. Moreover, the absence of a comprehensive metric to assess all different dimensions of image enhancement quality, such as detail preservation and visual naturalness, poses challenges in the training process and often results in outcomes that are not visually satisfactory.

3.3 Discussion

Some methods such as linear filters [34] over-smooth the image or blur sharp edges due to the lack of a suitable signal prior specifically tailored for PWP image reconstruction. Some other methods such as non-linear filters like bilateral filters, or variational denoising methods like TV [6] or GLR [18], cause a staircase effect by promoting PWC image reconstruction. On the other hand, some variational denoising methods such as [38, 41, 43, 47–51] mitigate these qualitative issues, but they suffer from high computational cost due to the usage of non-differentiable terms in their optimization objectives. In this thesis, we focus on using GGLR which overcomes all of these challenges to restore PWP images, using a convex and differentiable prior term that is amenable to fast optimization. Our primary focus is on restoring signals with PWP characteristics, such as depth images, or the linearly changing illumination component in contrast enhancement—scenarios where GGLR particularly excels.

Chapter 4

Proposed Method

In this chapter, we formulate an optimization objective with regularization terms specified using graph Laplacian matrices that promote signal smoothness with respect to defined graphs. Specifically, we employ a GGLR prior to promote PWP signal reconstruction, while simultaneously using a GLR prior to encourage signal continuity. Our approach allows practitioners to adjust the hyperparameters associated with the GLR term to either permit or prevent signal discontinuities in the final reconstructed signal. We first describe our method using a one-dimensional signal, then generalize it to 2D images processed in individual patches in the sequel. This investigation results in a single quadratic objective for PWP signal reconstruction, which can be efficiently solved in linear time by rewriting it as a system of linear equations and employing a *conjugate gradient* (CG) solver [42]. After describing our method, we discuss two practical applications: (*i*) low light image enhancement, and (*ii*) depth image denoising.

4.1 PWP Reconstruction of a one-dimensional Signal

We first formulate the problem of reconstructing a 1D signal using a GGLR prior, which promotes PWP signal reconstruction. Denote by $\mathbf{y} \in \mathbb{R}^N$ and $\mathbf{x} \in \mathbb{R}^N$ the length-N noisy observation and sought signal, respectively. Denote by $\mathcal{L} \in \mathbb{R}^{N \times N}$ a defined GNG Laplacian. We can now formulate an optimization for $\mathbf{x}$ given $\mathbf{y}$ as

$$\min_{\mathbf{x}} \quad \|\mathbf{x} - \mathbf{y}\|_2^2 + \mu_{\mathrm{g}} \mathbf{x}^\top \mathcal{L} \mathbf{x}, \tag{4.1}$$

where the first term accounts for fidelity, and the second prior term, with positive parameter $\mu_{\mathrm{g}} > 0$, promotes PWP signal reconstruction.

Given that the objective in (4.1) is convex ($\mathcal{L}$ is provably PSD) and differentiable with respect to $\mathbf{x}$, one can compute the solution $\mathbf{x}^*$ using the Karush-Kuhn-Tucker (KKT) [57] condition by taking the derivative with respect to $\mathbf{x}$ and setting it equal to zero, resulting in the following linear system to solve:

$$(\mathbf{I} + \mu_{\mathrm{g}} \mathcal{L})\, \mathbf{x} = \mathbf{y}. \tag{4.2}$$

4.1.1 Efficient Solvers using Sparse Conjugate Gradient

Looking at (4.2), the coefficient matrix $\mathbf{A} \triangleq \mathbf{I} + \mu_{\mathrm{g}} \mathcal{L}$ is provably positive definite (PD), and thus, (4.2) guarantees a unique solution that can be obtained using a linear system solver [58]. Further, $\mathbf{A}$ is the addition of an identity matrix and GNG Laplacian for a 1D signal, which is *pentadiagonal* [1], *i.e.*, matrix $\mathbf{A}$ has non-zeros only on the main diagonal plus two off-diagonals above and two below. The reason is because the

[1] $\mathcal{L} \in \mathbb{R}^{N \times N}$ being pentadiagonal means that for all $1 \leq i, j \leq N$ such that $\mathcal{L}_{i,j} = 0, \forall (i,j)$ s.t. $|i - j| > 2$.

graph Laplacian matrix $\bar{\mathbf{L}}$ for the gradient graph $\bar{\mathcal{G}}$ is *tridiagonal* for a line graph, and $\mathcal{L} = \mathbf{F}^\top \bar{\mathbf{L}} \mathbf{F}$, where $\mathbf{F}$ is a gradient operator that computes the difference between two rows of $\bar{\mathbf{L}}$. Thus, $\mathbf{A}$ is sparse with $\mathcal{O}(N)$ non-zero entries. This sparsity allows us to solve (4.2) using CG [42] in linear time, without performing matrix inversion.

As a brief overview, CG is an iterative descent algorithm: starting with an initial guess, CG iteratively updates the solution vector $\mathbf{x}$ by generating a sequence of conjugate directions along which it improves its current solutions until convergence. Each iteration involves either vector operations or matrix-vector multiplications with the coefficient matrix. Thus, CG can effectively leverage the sparsity of $\mathbf{A}$ in its matrix multiplications, resulting in each iteration taking $\mathcal{O}(N)$.

The convergence of CG is dependent on the matrix $\mathbf{A}$. Formally, given a solution $\mathbf{x}$ obtained from CG, if we want to reduce the error such that $\|\mathbf{x} - \mathbf{x}^*\|_2^2 \leq 2\varepsilon$ for $\varepsilon > 0$, the number of iterations required is $T \leq \mathcal{O}(\sqrt{\kappa(\mathbf{A})} \log(1/\varepsilon))$ where $\kappa(\cdot)$ is the condition number. Additionally, it can be shown[2] that $\kappa(\mathbf{A}) \leq \kappa(\mathcal{L})$, and with a reasonable parameter ε the total time complexity is $\mathcal{O}(TN)$ with $T \in \mathcal{O}(\sqrt{\kappa(\mathcal{L})} \log(1/\varepsilon))$ being independent of the signal length N, and thus the total algorithm is linear with respect to signal size. When the GNG Laplacian matrix is ill-conditioned ($\kappa(\mathcal{L})$ is high), we can employ a preconditioning matrix $\mathbf{M}$ and replace the system with $\mathbf{M}^{-1}(\mathbf{A}\mathbf{x} - \mathbf{y}) = 0$ such that $\kappa(\mathbf{M}^{-1}\mathbf{A})$ is well-conditioned. This can significantly improve the speed of the algorithm.

[2]The condition number is defined as the ratio of the largest to the smallest eigenvalue. If ν_1 and ν_N represent the smallest and largest eigenvalues of $\mathcal{L}$, respectively, then for $\mathbf{A}$, which has a rescaled and shifted eigenspectrum, the condition number can be expressed as $\kappa(\mathbf{A}) = \kappa(\mu_\mathrm{g}\mathcal{L} + \mathbf{I}) = \frac{\mu_\mathrm{g}\nu_N + 1}{\mu_\mathrm{g}\nu_1 + 1} \leq \frac{\nu_N}{\nu_1} = \kappa(\mathcal{L})$ for $\mu_\mathrm{g} \geq 0$.

4.1.2 Using GLR to Promote Continuity

As previously mentioned in Chapter 2, GGLR promotes PWP signal reconstruction by promoting PWC reconstruction in the gradient domain. However, GGLR does not enforce continuity in the pixel domain. Therefore, signals reconstructed by GGLR in (4.1) often exhibit discontinuities. Figure 4.1 shows a non-continuous PWP signal reconstruction using GGLR. While discontinuities are common in some practical scenarios such as natural and depth images (e.g., the boundary between a foreground object and background), they can be problematic when a continuous signal is desired. For instance, smoothness is important when the signal corresponds to a single smooth physical object in a depth map, or when the image is processed in patches, and the patch is small enough to exclude any edges or discontinuities in the signal.

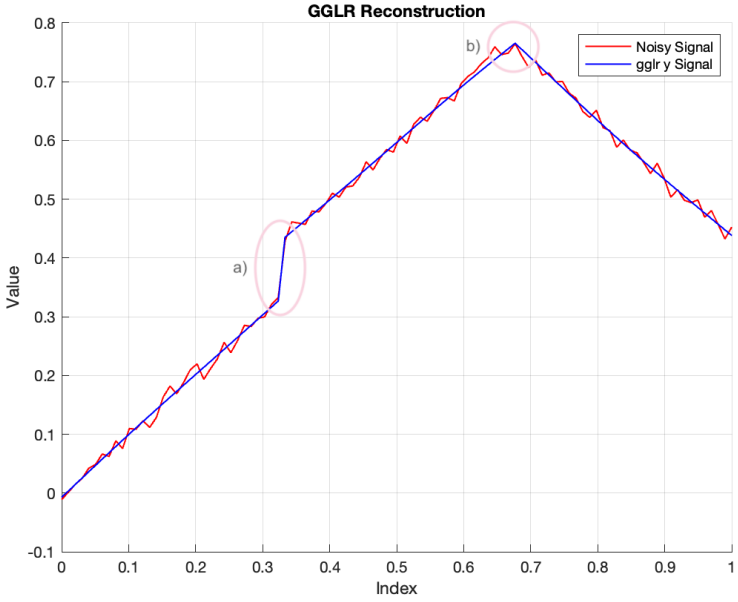


Figure 4.1: Reconstruction of a three-piece linear noisy signal using (4.1), showing a discontinuity at point (a) and continuity at point (b).

To address this, we introduce another regularization term in our optimization

objective to promote the continuity of the reconstructed signal. While low-pass filters are a simple way to impose continuity, we use a GLR term to keep the objective in (4.1) quadratic for ease of later optimization. Denote by $\mathbf{Q}$ the combinatorial graph Laplacian corresponding to a simple line graph with all edge weights equal to 1. Thus, we can write

$$\mathbf{x}^\top \mathbf{Q}\mathbf{x} = \sum_{n=1}^{N-1} (\mathbf{x}_{n+1} - \mathbf{x}_n)^2. \tag{4.3}$$

It is evident that discontinuities or large steps in the signals are penalized above, as they directly correspond to a significant difference between two consecutive signal samples. Thus, we add the GLR regularization with a coefficient μ_p into the objective to give us

$$\min_{\mathbf{x}} \quad \|\mathbf{x} - \mathbf{y}\|_2^2 + \mu_\mathrm{g} \mathbf{x}^\top \mathcal{L}\mathbf{x} + \mu_\mathrm{p} \mathbf{x}^\top \mathbf{Q}\mathbf{x}. \tag{4.4}$$

The objective is still quadratic, allowing us to solve this optimization problem via a linear system similar to (4.2). $\mathbf{Q}$ is sparse, because it is the graph Laplacian of a line graph, and thus the resulting system has a sparse coefficient matrix that we can solve efficiently using CG.

4.1.3 Spectral Re-ordering

Examining the GLR term in (4.3) more closely, we observe that while it promotes continuity, it also tends to favour constant signals, where all consecutive elements are equal (both quadratic priors evaluate to 0 for constant signals). This is overly restrictive when the desired signal has a more complicated shape. To address this, we modify $\mathbf{Q}$ to $\tilde{\mathbf{Q}}$ to target a specific signal shape. Specifically, we introduce the

concept of *mean-crossing*: the number of times a signal crosses its mean. A constant signal has 0 mean-crossings.

In general, when $\tilde{\mathbf{Q}}$ is PSD, the quadratic term $\mathbf{x}^\top \tilde{\mathbf{Q}}\mathbf{x}$ is non-negative and is minimized if and only if $\tilde{\mathbf{Q}}\mathbf{x} = \mathbf{0}$, which means $\mathbf{x}$ is a (unnormalized) first eigenvector of $\tilde{\mathbf{Q}}$ coresponding to first eigenvalue $\lambda_1 = 0$. Recall that *any* combinatorial graph Laplacian matrix $\mathbf{L}$ has a constant vector $\mathbf{c} = c\mathbf{1}$ as the first eigenvector corresponding to $\lambda_1 = 0$, *i.e.*, $\mathbf{Lc} = c\mathbf{L1} = \mathbf{0}$. Here, we increase the dimension of the eigen-subspace corresponding to $\lambda_1 = 0$ for matrix $\tilde{\mathbf{Q}}$ to two to allow a chosen high frequency $\mathbf{v}_s$ to also evaluate to zero, *i.e.*, $\tilde{\mathbf{Q}}\mathbf{v}_s = \tilde{\mathbf{Q}}\mathbf{c} = \mathbf{0}$.

Specifically, we perform *spectral re-ordering* as follows. Given a Laplacian $\mathbf{Q}$, with an eigen-pair $(\lambda_s, \mathbf{v}_s)$, $2 \leq s \leq N$, we define $\tilde{\mathbf{Q}} \triangleq \mathbf{Q} - \lambda_s \mathbf{v}_s \mathbf{v}_s^\top$. We can verify that $\mathbf{v}_s$ is now an eigenvector of $\tilde{\mathbf{Q}}$ corresponding to $\lambda_1 = 0$:

$$\begin{aligned} \tilde{\mathbf{Q}}\mathbf{v}_s &= (\mathbf{Q} - \lambda_s \mathbf{v}_s \mathbf{v}_s^\top)\mathbf{v}_s = \mathbf{Q}\mathbf{v}_s - \lambda_s \mathbf{v}_s (\mathbf{v}_s^\top \mathbf{v}_s) \\ &\overset{(a)}{=} \lambda_s \mathbf{v}_s - \lambda_s \mathbf{v}_s = \mathbf{0} \end{aligned} \tag{4.5}$$

where (a) is true since eigenvector $\mathbf{v}_s$ is unit-norm.

Suppose we know *a priori* that the target signal has $s-1$ mean-crossings, where $s \geq 2$. We can then identify eigenvector $\mathbf{v}_s$ of $\mathbf{Q}$ with the same number of mean-crossings as the target signal. Replacing $\mathbf{Q}$ in the quadratic objective of GLR with $\tilde{\mathbf{Q}}$ will not only promote continuity, but also encourage the signal to have the same

target number of mean-crossings. We can now redefine the following objective:

$$\begin{aligned} \min_{\mathbf{x},s} \quad & \|\mathbf{x}-\mathbf{y}\|_2^2 + \mu_{\mathrm{g}}\mathbf{x}^\top \mathcal{L}\mathbf{x} + \mu_{\mathrm{p}}\mathbf{x}^\top(\mathbf{Q}-\lambda_s \mathbf{v}_s\mathbf{v}_s^\top)\mathbf{x} \\ \text{s.t.} \quad & \mathbf{Q}\mathbf{v}_s = \lambda_s \mathbf{v}_s. \end{aligned} \tag{4.6}$$

In practice, we do not optimize s directly. Instead, we use a heuristic to estimate s as follows. First, we perform pre-denoising on $\mathbf{y}$ to obtain $\hat{\mathbf{y}}$, using an off-the-shelf algorithm such as BM3D [43]. Next, we count the mean-crossings of $\hat{\mathbf{y}}$, and based on this count, we identify the appropriate eigenvector $\mathbf{v}_s$ with the same number of mean crossings. We then select this eigenvalue and eigenvector for our spectral reordering. This method removes s as a variable from the objective, making it quadratic and tractable again.

Figure 4.2 illustrates a continuous PWP signal reconstructed using (4.1) and (4.6) with the same parameters. While GGLR results in a PWP reconstruction signal, the presence of noise causes discontinuities. However, adding a GLR term with $s = 2$ to match the number of mean crossings of the signal results in a continuous reconstruction.

Having chosen s, (4.6) is still convex and can be solved via a linear system:

$$\left(\mathbf{I} + \mu_{\mathrm{g}}\mathcal{L} + \mu_{\mathrm{p}}(\mathbf{Q}-\lambda_s \mathbf{v}_s\mathbf{v}_s^\top)\right)\mathbf{x} = \mathbf{y} \tag{4.7}$$

However, unlike before, the linear system would be non-sparse after applying the spectral re-ordering, since the rank-1 term $\mathbf{v}_s\mathbf{v}_s^\top$ is a dense matrix and cannot be solved in $\mathcal{O}(N)$ using only CG. In the following, we employ auxiliary variables and define a new optimization objective that can be solved in linear time using the *Alternating*

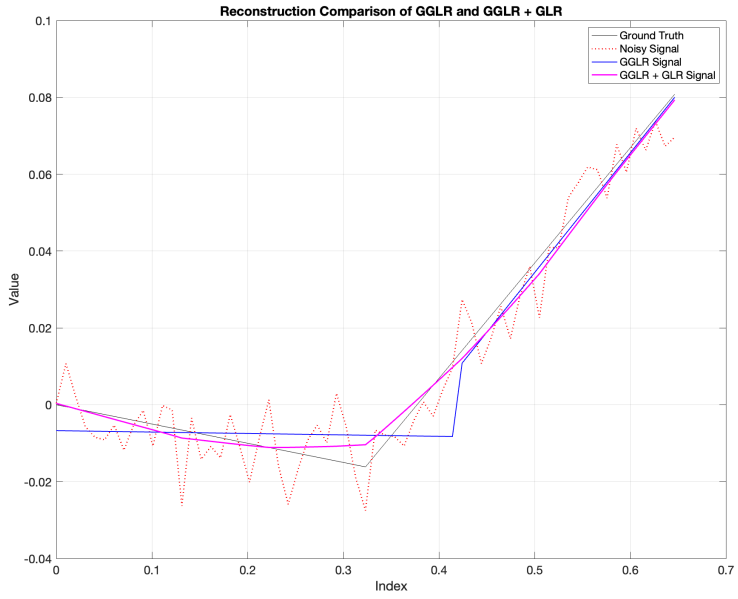


Figure 4.2: Reconstruction of a two-piece continuous linear signal using: i) GGLR and ii) GGLR combined with signal-independent GLR and spectral reordering with $s = 2$ for a continuous PWP reconstruction.

Direction Method of Multipliers (ADMM) method [59].

4.1.4 A Linear-time Algorithm using ADMM

ADMM is an algorithm designed to solve optimization problems by breaking them into smaller, more manageable subproblems. In our case, as we will see, we break the optimization problem into two sub-problems by defining auxiliary variables.

First, we briefly cover the general formulation of ADMM. Consider the following optimization problem:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & h(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{H}\mathbf{x} + \mathbf{G}\mathbf{z} = \mathbf{c}, \end{aligned} \tag{4.8}$$

where $\mathbf{x} \in \mathbb{R}^N$, and $\mathbf{z} \in \mathbb{R}^K$ are vectors; $h : \mathbb{R}^N \to \mathbb{R}$ and $g : \mathbb{R}^K \to \mathbb{R}$ are convex

functions; $\mathbf{H} \in \mathbb{R}^{M\times N}$ and $\mathbf{G} \in \mathbb{R}^{M\times K}$ are matrices; and $\mathbf{c} \in \mathbb{R}^M$ is a vector specifying a linear constraint. The unconstrained augmented Lagrangian version for this problem is

$$\mathfrak{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = h(\mathbf{x}) + g(\mathbf{z}) + \mathbf{u}^\top(\mathbf{H}\mathbf{x} + \mathbf{G}\mathbf{z} - \mathbf{c}) + \frac{\rho}{2}\|\mathbf{H}\mathbf{x} + \mathbf{G}\mathbf{z} - \mathbf{c}\|_2^2, \tag{4.9}$$

where $\mathbf{u} \in \mathbb{R}^M$ is the Lagrange multiplier vector, and $\rho > 0$ is a scalar parameter.

ADMM minimizes this unconstrained objective by iteratively miminizing one variable at a time while holding the others fixed, until solution convergence. In particular, it has the following iterative steps:

$$\mathbf{x}^{(t+1)} = \arg\min_{\mathbf{x}} \left(h(\mathbf{x}) + (\mathbf{u}^{(t)})^\top \mathbf{H}\mathbf{x} + \frac{\rho}{2} \left\| \mathbf{H}\mathbf{x} + \mathbf{G}\mathbf{z}^{(t)} - \mathbf{c} \right\|_2^2 \right) \tag{4.10}$$

$$\mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z}} \left(g(\mathbf{z}) + (\mathbf{u}^{(t)})^\top \mathbf{G}\mathbf{z} + \frac{\rho}{2} \left\| \mathbf{H}\mathbf{x}^{(t+1)} + \mathbf{G}\mathbf{z} - \mathbf{c} \right\|_2^2 \right) \tag{4.11}$$

$$\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} + \rho \cdot (\mathbf{H}\mathbf{x}^{(t+1)} + \mathbf{G}\mathbf{z}^{(t+1)} - \mathbf{c}) \tag{4.12}$$

where we call (4.10) "$\mathbf{x}$-update", (4.11) "$\mathbf{z}$-update", and (4.12) "Lagrange-multiplier-update".

To employ ADMM, we define an auxiliary variable $\mathbf{z}$ and add the condition $\mathbf{x} = \mathbf{z}$ to our optimization problem, effectively turning our unconstrained optimization problem (4.6) with s fixed into a constrained optimization problem, *i.e.*,

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{x} - \mathbf{y}\|_2^2 + \mu_\mathrm{g}\mathbf{x}^\top \mathcal{L}\mathbf{x} + \mu_\mathrm{p}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mu_\mathrm{p}\mathbf{z}^\top \lambda_s \mathbf{v}_s \mathbf{v}_s^\top \mathbf{z} \\ \text{subject to} \quad & \mathbf{x} = \mathbf{z}. \end{aligned} \tag{4.13}$$

Rewriting the above into its unconstrained augmented Lagrangian form, we get

$$\begin{aligned}\min_{\mathbf{x},\mathbf{z}} \quad & \|\mathbf{x}-\mathbf{y}\|_2^2 + \mu_{\mathrm{g}}\mathbf{x}^\top\mathcal{L}\mathbf{x} + \mu_{\mathrm{p}}\mathbf{x}^\top\mathbf{Q}\mathbf{x} - \mu_{\mathrm{p}}\mathbf{z}^\top\lambda_s\mathbf{v}_s\mathbf{v}_s^\top\mathbf{z} \\ & + \mathbf{u}^\top(\mathbf{x}-\mathbf{z}) + \frac{\rho}{2}\|\mathbf{x}-\mathbf{z}\|_2^2. \end{aligned} \tag{4.14}$$

Following the ADMM frmework, we can solve for $\mathbf{x}$ and $\mathbf{z}$ and update $\mathbf{u}$ iteratively as follows:

$$\mathbf{x}^{(t+1)} = \arg\min_{\mathbf{x}} \left(\|\mathbf{x}-\mathbf{y}\|_2^2 + \mu_{\mathrm{g}}\mathbf{x}^\top\mathcal{L}\mathbf{x} + \mu_{\mathrm{p}}\mathbf{x}^\top\mathbf{Q}\mathbf{x} + (\mathbf{u}^{(t)})^\top\mathbf{x} + \frac{\rho}{2}\left\|\mathbf{x}-\mathbf{z}^{(t)}\right\|_2^2 \right) \tag{4.15}$$

$$\mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z}} \left(-\mu_{\mathrm{p}}\mathbf{z}^\top\lambda_s\mathbf{v}_s\mathbf{v}_s^\top\mathbf{z} - (\mathbf{u}^{(t)})^\top\mathbf{z} + \frac{\rho}{2}\left\|\mathbf{x}^{(t+1)}-\mathbf{z}\right\|_2^2 \right) \tag{4.16}$$

$$\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} + \rho\cdot(\mathbf{x}^{(t+1)} - \mathbf{z}^{(t+1)}) \tag{4.17}$$

We note that the $\mathbf{x}$-update and the $\mathbf{z}$-update phase in ADMM involve optimizing a differentiable quadratic objective. This optimization can be achieved by applying the KKT condition as done previously. Specifically, we take the derivative of an objective function with respect to the variable we aim to minimize and set it to zero. For each sub-problem, performing this computation yields:

$$\left((2+\rho)\mathbf{I} + 2\mu_{\mathrm{g}}\mathcal{L} + 2\mu_{\mathrm{p}}\mathbf{Q}\right)\mathbf{x}^{(t+1)} = \left(\rho\mathbf{z}^{(t)} + 2\mathbf{y} - \mathbf{u}^{(t)}\right) \tag{4.18}$$

$$\mathbf{z}^{(t+1)} = \left(-2\mu_{\mathrm{p}}\lambda_s\mathbf{v}_s\mathbf{v}_s^\top + \rho\mathbf{I}\right)^{-1}\left(\mathbf{u}^{(t)} + \rho\mathbf{x}^{(t+1)}\right) \tag{4.19}$$

$$\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} + \rho\left(\mathbf{x}^{(t+1)} - \mathbf{z}^{(t+1)}\right) \tag{4.20}$$

As done previously, (4.18) involves solving a sparse linear system that can be done in

linear time using CG. (4.19) on the other hand, is dense but contains matrix inversion with a rank-1 update. Thus, instead of computing a matrix inversion, we employ the Sherman-Morrison formula [60] to derive the following:

$$\mathbf{z}^{(t+1)} = \left(-2\mu_{\mathrm{p}}\lambda_s \mathbf{v}_s \mathbf{v}_s^\top + \rho \mathbf{I}\right)^{-1} \left(\mathbf{u}^{(t)} + \rho \mathbf{x}^{(t+1)}\right) \tag{4.21}$$

$$= \left(\frac{1}{\rho}\mathbf{I} + \frac{2\mu_{\mathrm{p}}\lambda_s}{\rho(\rho - 2\mu_{\mathrm{p}}\lambda_s)} \mathbf{v}_s \mathbf{v}_s^\top\right) \left(\mathbf{u}^{(t)} + \rho \mathbf{x}^{(t+1)}\right) \tag{4.22}$$

$$= \frac{1}{\rho}\mathbf{u}^{(t)} + \mathbf{x}^{(t+1)} + \frac{2\mu_{\mathrm{p}}\lambda_s}{\rho(\rho - 2\mu_{\mathrm{p}}\lambda_s)} \mathbf{v}_s^\top \left(\mathbf{u}^{(t)} + \rho \mathbf{x}^{(t+1)}\right) \mathbf{v}_s. \tag{4.23}$$

In details, We first calculate the dot product of vectors $\mathbf{v}_s$ and $(\mathbf{u}^{(t)} + \rho \mathbf{x}^{(t+1)})$ resulting in a scalar. This step requires $\mathcal{O}(N)$ operations. Then, we scale the obtained dot product with $\frac{2\mu_{\mathrm{p}}\lambda_s}{\rho(\rho - 2\mu_{\mathrm{p}}\lambda_s)}$ resulting in another scalar, which we can then multiply with vector $\mathbf{v}_s$, again in $\mathcal{O}(N)$ operations. Thus, computing $\mathbf{z}^{(t+1)}$ in (4.23) can be done in $\mathcal{O}(N)$ operations. Finally, (4.20) is also be computed in linear time trivially.

In summary, introducing auxiliary variables and employing ADMM enables a linear-time algorithm: it takes multiple iterations, and in each iteration, the parameters are updated via either a sparse CG or a set of linear vector operations. Finally, ADMM takes a total of T steps to converge, where T is not a function of N, then our algorithm has a complexity of $\mathcal{O}(TN)$, which is linear.

4.2 Adapting the Algorithm for Images

To adapt our algorithm for 2D images, we first divide our image into patches of size $N \times N$, and then perform our reconstruction algorithm on pixel rows and columns of each patch. Denote the vectorized signal for each patch by $\mathbf{y} \in \mathbb{R}^{N^2}$, obtained by concatenating the N columns of the patch, each of size N. Denote by $\mathbf{x} \in \mathbb{R}^{N^2}$

the target reconstructed signal, which is vectorized like $\mathbf{y}$. Reconstruction occurs sequentially, one patch at a time, from top to bottom and left to right.

4.2.1 Patch Laplacians and Selection Matrices

Instead of a single GNG Laplacian $\mathcal{L}$ and a Laplacian $\mathbf{Q}$ for the entire signal, we need to define a GNG Laplacian and Laplacian for each row and column. Thus, for each $1 \leq n \leq N$, we define row and column GNG Laplacians as $\mathcal{L}_{r,n}$ and $\mathcal{L}_{r,n}$, respectively, and define row and column (signal-independent) Laplacians as $\mathbf{Q}_{r,n}$ and $\mathbf{Q}_{c,n}$, respectively. Furthermore, for spectral re-ordering, we denote the s-th eigenvalue of $\mathbf{Q}_{r,n}$ and $\mathbf{Q}_{c,n}$ with $\lambda_{r,n,s}$ and $\lambda_{c,n,s}$, and the s-th eigenvector with $\mathbf{v}_{r,n,s}$ and $\mathbf{v}_{c,n,s}$.

To define regularization terms, we need to access the n-th pixel row and n-th pixel column of the patch. Thus, we also define row/column selection matrices $\mathbf{S}_{r,n} \in \{0,1\}^{N \times N^2}$ and $\mathbf{S}_{c,n} \in \{0,1\}^{N \times N^2}$, where $\mathbf{S}_{r,n}$ and $\mathbf{S}_{c,n}$ are all N-by-N^2 matrices such that $\mathbf{S}_{c,n}\mathbf{x}$ effectively picks the pixels corresponding to the n-th column of $\mathbf{x}$ and $\mathbf{S}_{r,n}\mathbf{x}$ picks the pixels in the n-th row of $\mathbf{x}$. For example, the regularization term $\mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathcal{L}_{r,n} \mathbf{S}_{r,n}\mathbf{x}$ enforces PWP on the n-th row of the patch, and the term $\mathbf{x}^\top \mathbf{S}_{c,n}^\top (\mathbf{Q}_{c,n} - \lambda_{c,n,s}\mathbf{v}_{c,n,s}\mathbf{v}_{c,n,s}^\top)\mathbf{S}_{c,n}\mathbf{x}$ enforces continuity on the n-th column of the patch using spectrally re-ordered GLR.

As an illustrative example, $\mathbf{S}_{c,1}$, $\mathbf{S}_{r,2}$, and $\mathbf{B}$ for a 2-by-2 patch are:

$$\mathbf{S}_{c,1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \mathbf{S}_{r,2} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.24}$$

4.2.2 Patch Based Objective

We now formulate the patch-based optimization problem by incorporating all the concepts discussed in the previous sections. Similar to before, we use μ_g and μ_p as importance weights for our GGLR and GLR priors, leading to the following:

$$\begin{aligned}\min_{\mathbf{x}} \quad & \|\mathbf{x}-\mathbf{y}\|_2^2 \\ & + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathcal{L}_{r,n} \mathbf{S}_{r,n} \mathbf{x} + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{c,n}^\top \mathcal{L}_{c,n} \mathbf{S}_{c,n} \mathbf{x} \\ & + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{r,n}^\top (\mathbf{Q}_{r,n} - \lambda_{r,n,s} \mathbf{v}_{r,n,s} \mathbf{v}_{r,n,s}^\top)\, \mathbf{S}_{r,n} \mathbf{x} \qquad (4.25) \\ & + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{c,n}^\top (\mathbf{Q}_{c,n} - \lambda_{c,n,s} \mathbf{v}_{c,n,s} \mathbf{v}_{c,n,s}^\top) \mathbf{S}_{c,n} \mathbf{x} \qquad (4.26)\end{aligned}$$

where the first term is the fidelity term, the second and third terms promote PWP signal reconstruction via GGLR, and the fourth and fifth terms ensure both continuity and compliance with a specific mean-crossing using GLR.

4.2.3 Adapting ADMM

Similar to the solution in (4.6), we define auxiliary variables. However, instead of a single auxiliary vector $\mathbf{z}$, we define a set of row/column auxiliary variables $\mathbf{z}_{c,n} \in \mathbb{R}^N$ and $\mathbf{z}_{r,n} \in \mathbb{R}^N$ for $1 \leq n \leq N$ with the constraint $\mathbf{z}_{c,n} = \mathbf{S}_{c,n}\mathbf{x}$ and $\mathbf{z}_{r,n} = \mathbf{S}_{r,n}\mathbf{x}$, respectively. This leads to the following constrained optimization problem that has the same optimal solution as (4.25):

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & \|\mathbf{x}-\mathbf{y}\|_2^2 \\
& + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathcal{L}_{r,n} \mathbf{S}_{r,n} \mathbf{x} + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{c,n}^\top \mathcal{L}_{c,n} \mathbf{S}_{c,n} \mathbf{x} \\
& + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathbf{Q}_{r,n} \mathbf{S}_{r,n} \mathbf{x} + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{c,n}^\top \mathbf{Q}_{c,n} \mathbf{S}_{c,n} \mathbf{x} \\
& - \sum_{n=1}^{N} \mu_p \lambda_{r,n,s} \mathbf{z}_{r,n}^\top \mathbf{v}_{r,n,s} \mathbf{v}_{r,n,s}^\top \mathbf{z}_{r,n} \\
& - \sum_{n=1}^{N} \mu_p \lambda_{c,n,s} \mathbf{z}_{c,n}^\top \mathbf{v}_{c,n,s} \mathbf{v}_{c,n,s}^\top \mathbf{z}_{c,n} \\
\text{Subject to} \quad & \forall_{1 \le n \le N} : \mathbf{S}_{c,n} \cdot \mathbf{x} = \mathbf{z}_{c,n} \\
& \forall_{1 \le n \le N} : \mathbf{S}_{r,n} \cdot \mathbf{x} = \mathbf{z}_{r,n}
\end{aligned}
\tag{4.27}
$$

As done previously, rewriting (4.27) with the unconstrained objective in ADMM

yields the following:

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & \|\mathbf{x}-\mathbf{y}\|_2^2 \\
& + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathcal{L}_{r,n} \mathbf{S}_{r,n} \mathbf{x} + \sum_{n=1}^{N} \mu_g \mathbf{x}^\top \mathbf{S}_{c,n}^\top \mathcal{L}_{c,n} \mathbf{S}_{c,n} \mathbf{x} \\
& + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{r,n}^\top \mathbf{Q}_{r,n} \mathbf{S}_{r,n} \mathbf{x} + \sum_{n=1}^{N} \mu_p \mathbf{x}^\top \mathbf{S}_{c,n}^\top \mathbf{Q}_{c,n} \mathbf{S}_{c,n} \mathbf{x} \\
& - \sum_{n=1}^{N} \mu_p \lambda_{r,n,s} \mathbf{z}_{r,n}^\top \mathbf{v}_{r,n,s} \mathbf{v}_{r,n,s}^\top \mathbf{z}_{r,n} \\
& - \sum_{n=1}^{N} \mu_p \lambda_{c,n,s} \mathbf{z}_{c,n}^\top \mathbf{v}_{c,n,s} \mathbf{v}_{c,n,s}^\top \mathbf{z}_{c,n} \\
& + \sum_{n=1}^{N} \mathbf{u}_{r,n}^\top (\mathbf{S}_{r,n}\mathbf{x} - \mathbf{z}_{r,n}) + \sum_{n=1}^{N} \mathbf{u}_{c,n}^\top (\mathbf{S}_{c,n}\mathbf{x} - \mathbf{z}_{c,n}) \\
& + \frac{\rho}{2} \sum_{n=1}^{N} \|\mathbf{S}_{r,n}\mathbf{x} - \mathbf{z}_{r,n}\|_2^2 + \frac{\rho}{2} \sum_{n=1}^{N} \|\mathbf{S}_{c,n}\mathbf{x} - \mathbf{z}_{c,n}\|_2^2
\end{aligned}
\tag{4.28}
$$

Adapting the ADMM optimization framework, we define the sequence $\{\mathbf{x}^{(t)}\}$ and sequences $\{\mathbf{z}_{r,n}^{(t)}\}$, $\{\mathbf{z}_{c,n}^{(t)}\}$, $\{\mathbf{u}_{r,n}^{(t)}\}$, and $\{\mathbf{u}_{c,n}^{(t)}\}$ for all $1 \leq n \leq N$, which we iteratively update via optimization of sub-problems addressing one variable at a time. For each update, given differentiability and the quadratic objective, we can take the derivative with respect to the optimization variable and set it to zero, giving us a linear system to solve.

For the "x-update" function, for example, taking the derivative yields:

$$
\begin{aligned}
&2(\mathbf{x}-\mathbf{y}) \\
&+2\sum_{n=1}^{N}\mu_g\mathbf{S}_{r,n}^{\top}\mathcal{L}_{r,n}\mathbf{S}_{r,n}\mathbf{x}+2\sum_{n=1}^{N}\mu_g\mathbf{S}_{c,n}^{\top}\mathcal{L}_{c,n}\mathbf{S}_{c,n}\mathbf{x} \\
&+2\sum_{n=1}^{N}\mu_p\mathbf{S}_{r,n}^{\top}\mathbf{Q}_{r,n}\mathbf{S}_{r,n}\mathbf{x}+2\sum_{n=1}^{N}\mu_g\mathbf{S}_{c,n}^{\top}\mathbf{Q}_{c,n}\mathbf{S}_{c,n}\mathbf{x} \\
&-\sum_{n=1}^{N}\mathbf{S}_{r,n}^{\top}\mathbf{u}_{r,n}^{(t)}-\sum_{n=1}^{N}\mathbf{S}_{c,n}^{\top}\mathbf{u}_{c,n}^{(t)} \\
&+\rho\sum_{n=1}^{N}\mathbf{S}_{r,n}^{\top}(\mathbf{S}_{r,n}\mathbf{x}-\mathbf{z}_{r,n}^{(t)})+\rho\sum_{n=1}^{N}\mathbf{S}_{c,n}^{\top}(\mathbf{S}_{c,n}\mathbf{x}-\mathbf{z}_{c,n}^{(t)})=0
\end{aligned}
\tag{4.29}
$$

Therefore, the ADMM update step involves solving the linear system entailed by (4.29). Which is defined as follows:

$$
\begin{aligned}
\mathbf{A} := \quad & \Bigg(2\mathbf{I}+\sum_{n=1}^{N}\mathbf{S}_{r,n}^{\top}\left[2\mu_g\mathcal{L}_{r,n}+2\mu_p\mathbf{Q}_{r,n}+\rho\mathbf{I}\right]\mathbf{S}_{r,n}+ \\
& +\sum_{n=1}^{N}\mathbf{S}_{c,n}^{\top}\left[2\mu_g\mathcal{L}_{c,n}+2\mu_p\mathbf{Q}_{c,n}+\rho\mathbf{I}\right]\mathbf{S}_{c,n}\Bigg)
\end{aligned}
\tag{4.30}
$$

$$
\Rightarrow\ \mathbf{A}\mathbf{x}^{(t+1)} := \quad \left(2\mathbf{y}+\sum_{n=1}^{N}\mathbf{S}_{r,n}^{\top}\left[\rho\mathbf{z}_{r,n}^{(t)}-\mathbf{u}_{r,n}^{(t)}\right]+\sum_{n=1}^{N}\mathbf{S}_{c,n}^{\top}\left[\rho\mathbf{z}_{c,n}^{(t)}-\mathbf{u}_{c,n}^{(t)}\right]\right) \tag{4.31}
$$

Similar to before, it can be shown that $\mathbf{A}$ is sparse and positive definite: $\mathbf{A}$ is the summation of some PSD and positive definite matrices, and therefore, is positive definite using Weyl's inequality. That said, the linear system has a unique solution and can be solved efficiently with sparse CG.

For z-update, we can update each component individually. In other words, we can compute the gradient with respect to $\mathbf{z}_{r,n}$ and $\mathbf{z}_{c,n}$ independently for all $1 \leq n \leq N$.

Taking the derivative with respect to $\mathbf{z}_{r,n}$ and $\mathbf{z}_{c,n}$ will give us the following (repeat for all $1 \leq n \leq N$):

$$
\begin{aligned}
&-2\mu_p\lambda_{r,n,s}\mathbf{v}_{r,n,s}\mathbf{v}_{r,n,s}^\top\mathbf{z}_{r,n} - \mathbf{u}_{r,n}^{(t)} - \rho(\mathbf{S}_{r,n}\mathbf{x}^{(t+1)} - \mathbf{z}_{r,n}) = 0 \\
\Rightarrow\ &\left[\rho\mathbf{I} - 2\mu_p\lambda_{r,n,s}\mathbf{v}_{r,n,s}\mathbf{v}_{r,n,s}^\top\right]\mathbf{z}_{r,n} = \mathbf{v}_{r,n,s} + \rho\mathbf{S}_{r,n}\mathbf{x} \\
\Rightarrow\ &\mathbf{z}_{r,n}^{(t+1)} = \tfrac{1}{\rho}\mathbf{u}_{r,n}^{(t)} + \mathbf{S}_{r,n}\mathbf{x}^{(t+1)} + \tfrac{2\mu_p\lambda_{r,n,s}}{\rho^2 - 2\rho\mu_p\lambda_{r,n,s}}\left[\mathbf{v}_{r,n,s}^\top\left(\mathbf{v}_{r,n,s} + \rho\mathbf{S}_{r,n}\mathbf{x}^{(t+1)}\right)\right]\mathbf{v}_{r,n,s} \qquad (4.32) \\
\Rightarrow\ &\left[\rho\mathbf{I} - 2\mu_p\lambda_{c,n,s}\mathbf{v}_{c,n,s}\mathbf{v}_{c,n,s}^\top\right]\mathbf{z}_{c,n} = \mathbf{u}_{c,n}^{(t)} + \rho\mathbf{S}_{c,n}\mathbf{x}^{(t+1)} \qquad \text{Similar to row updates} \\
\Rightarrow\ &\mathbf{z}_{c,n}^{(t+1)} = \tfrac{1}{\rho}\mathbf{u}_{c,n}^{(t)} + \mathbf{S}_{c,n}\mathbf{x}^{(t+1)} + \tfrac{2\mu_p\lambda_{c,n,s}}{\rho^2 - 2\rho\mu_p\lambda_{c,n,s}}\left[\mathbf{v}_{c,n,s}^\top(\mathbf{u}_{c,n}^{(t)} + \rho\mathbf{S}_{c,n}\mathbf{x}^{(t+1)})\right]\mathbf{v}_{c,n,s}. \qquad (4.33)
\end{aligned}
$$

Note that the final equations in (4.32) and (4.33) are obtained using the Sherman-Morrison formula. Finally, the Lagrange-multiplier-updates are

$$\mathbf{u}_{r,n}^{(t+1)} := \mathbf{u}_{r,n}^{(t)} + \rho(\mathbf{S}_{r,n}\mathbf{x}^{(t+1)} - \mathbf{z}_{r,n}^{(t+1)}) \qquad \forall 1 \leq n \leq N \qquad (4.34)$$

$$\mathbf{u}_{c,n}^{(t+1)} := \mathbf{u}_{c,n}^{(t)} + \rho(\mathbf{S}_{c,n}\mathbf{x}^{(t+1)} - \mathbf{z}_{c,n}^{(t+1)}) \qquad \forall 1 \leq n \leq N \qquad (4.35)$$

All of the update functions in (4.31), (4.32), (4.33), (4.34), and (4.35) can be done in $\mathcal{O}(N^2)$ which is linear with respect to the size of the patch.

4.3 Depth Image Denoising

Our constructed ccontinuous PWP prior is particularly well-suited for depth images due to the planar characteristics of object surfaces. Consequently, we can directly apply the algorithm described in Section 4.2 to perform depth image denoising. However, we will go over a few practical considerations in this section.

To ensure that our reconstruction accurately captures the boundaries of physical

objects in the image, we perform a preprocessing step using edge detection to detect signal discontinuities. While some datasets, such as [2], provide a discontinuity map, it can also be efficiently generated using the Canny edge detection method [61]. After signal discontinuity detection, we remove edges of path graphs that fall between boundaries. This converts each line graph into a set of multiple line graphs, each corresponding to a smooth, continuous signal. This approach significantly enhances the performance of our method.

4.4 Low Light Image Contrast Enhancement and Denoising

To process low-light images for contrast enhancement and denoising, similarly done in [28, 47, 48], we mathematically interpret the Retinex theory [5] to mean that a ground-truth N-by-N image patch, vectorized to strictly positive $\mathbf{x} \in \mathbb{R}_+^{N^2}$, is a point-by-point multiplication of strictly positive illumination and reflectance components, $\mathbf{l}, \mathbf{r} \in \mathbb{R}_+^{N^2}$. Specifically, the image formation model for observation $\mathbf{y} \in \mathbb{R}_+^{N^2}$ is

$$\mathbf{y} = \mathbf{l} \odot \mathbf{r} + \mathbf{z}, \tag{4.36}$$

where $\mathbf{z}$ is a zero-mean additive Gaussian noise. Reflectance $\mathbf{r}$ depends only on the surface properties of physical objects and is known to be piecewise smooth or PWC, and we reconstruct it using SDGLR as signal prior. In contrast, illumination $\mathbf{l}$ varies less drastically than $\mathbf{r}$; we model $\mathbf{l}$ it to be continuous PWP and we reconstruct it using GGLR as signal prior.

4.4.1 Initialization of Illumination and Reflectance

Our method requires initialization of the illumination and reflectance components, $\mathbf{l}$ and $\mathbf{r}$, of a N-by-N pixel patch, after which one component is optimized while the other is held fixed. To initialize $\mathbf{l}$, we compute the blurred V component of an input image in the HSV colour space using a Gaussian filter with a standard deviation of 5. We then initialize $\mathbf{r}$ by performing a point-by-point division of the image's intensity component ($\mathbf{y}$) by $\mathbf{l}$.

4.4.2 Computation of Reflectance

Reflectance Graph Construction We construct a graph $\mathcal{G}_{r,k}$ ($\mathcal{G}_{c,k}$) by connecting pixels i and j in row k (column k) with edge weight $w_{i,j}^{r,k}$ ($w_{i,j}^{c,k}$) signal dependently, similar to (2.2):

$$w_{i,j}^{r,k} = \exp\left(\frac{-(r_i - r_j)^2}{\sigma_r^2}\right), \tag{4.37}$$

where r_i is the reflectance intensity of pixel i, and σ_r is a user-defined parameter. The corresponding signal-dependent graph Laplacian matrix is defined as $\mathbf{L}_{r,k} \triangleq \mathrm{diag}(\mathbf{W}_{r,k}\mathbf{1}) - \mathbf{W}_{r,k} \in \mathbb{R}^{N\times N}$. We use $\mathbf{L}_{c,k}$ for the graph Laplacian matrix of the k-th column of a target patch.

Reflectance Optimization Given illumination $\mathbf{l}$ for a N^2-pixel patch, we compute reflectance $\mathbf{r}$ by minimizing an unconstrained convex quadratic objective:

$$\min_{\mathbf{r}} \|\mathbf{y} - \mathrm{diag}(\mathbf{l})\mathbf{r}\|_2^2 + \mu_p \sum_{k=1}^{N} \mathbf{r}^\top \mathbf{S}_{r,k}^\top \mathbf{L}_{r,k} \mathbf{S}_{r,k}\mathbf{r} + \mu_p \sum_{k=1}^{N} \mathbf{r}^\top \mathbf{S}_{c,k}^\top \mathbf{L}_{c,k} \mathbf{S}_{c,k}\mathbf{r}, \tag{4.38}$$

where μ_p is a parameter that trades off the fidelity term and the GLRs. The solution $\mathbf{r}^*$ to (4.38) is obtained by solving a linear system:

$$\left(\text{diag}^2(\mathbf{l}) + \mu_p \sum_{k=1}^{N} (\mathbf{S}_{r,k}^\top \mathbf{L}_{r,k} \mathbf{S}_{r,k} + \mathbf{S}_{c,k}^\top \mathbf{L}_{c,k} \mathbf{S}_{c,k})\right) \mathbf{r}^* = \text{diag}(\mathbf{l})\mathbf{y}, \tag{4.39}$$

which can be computed via CG [42] in linear time without matrix inversion.

4.4.3 Computation of Illumination

Based on the PWP assumption on illumination, we regularize $\mathbf{l}$ using GGLR [33] as explained in Section 4.1. This requires the construction of a gradient graph for each row / column of pixels in a target image patch, on which we define a GLR. Then, we map the gradient GLR back to the pixel domain as GGLR for optimization.

Illumination Graph Construction Given $\mathbf{S}_{r,k}\mathbf{l}$, and gradient operator F defined in (2.3), we first compute horizontal gradient $\mathbf{g} = \mathbf{F}\mathbf{S}_{r,k}\mathbf{l} \in \mathbb{R}^{N-1}$. We then construct a gradient graph $\bar{\mathcal{G}}_{r,l}$ to connect $N-1$ gradients in $\mathbf{g}$ as follows. We connect gradients i and j with edge weight $\bar{w}_{i,j}^{r,k}$ similar to (2.4):

$$\bar{w}_{i,j}^{r,k} = \exp\left(\frac{-(g_i - g_j)^2}{\sigma_l^2}\right). \tag{4.40}$$

Collection of edge weights $\{\bar{w}_{i,j}^{r,k}\}$ (4.40) thus defines adjacency matrix $\bar{\mathbf{W}}_{r,k}$ and subsequent gradient graph Laplacian $\bar{\mathbf{L}}_{r,k} \triangleq \text{diag}(\bar{\mathbf{W}}_{r,k}\mathbf{1}) - \bar{\mathbf{W}}_{r,k}$. We use $\bar{\mathbf{L}}_{c,k}$ for graph Laplacian matrix of gradient graph $\bar{\mathcal{G}}_{c,k}$ for the k-th column of a target patch.

Illumination Optimization Given $\bar{\mathbf{L}}_{r,k}$, we define GLR for gradient $\mathbf{g}$:

$$\mathbf{g}^\top \bar{\mathbf{L}}_{r,k} \mathbf{g} = \mathbf{l}^\top \mathbf{S}_{r,k}^\top \underbrace{\mathbf{F}^\top \bar{\mathbf{L}}_{r,k} \mathbf{F}}_{\mathcal{L}_{r,k}} \mathbf{S}_{r,k} \mathbf{l} \tag{4.41}$$

where $\mathcal{L}_{r,k} = \mathbf{F}^\top \bar{\mathbf{L}}_l \mathbf{F}$ is a GNG Laplacian. $\mathcal{L}_{r,k}$ corresponds to a graph $\mathcal{G}_{r,k}^g$ connecting N illumination pixels. $\mathbf{l}^\top \mathbf{S}_{r,k}^\top \mathcal{L}_{r,k} \mathbf{S}_{r,k} \mathbf{l}$ is a GGLR for the k-th pixel row of the target illumination patch.

Given reflectance $\mathbf{r}$, similar to (4.1), we compute illumination $\mathbf{l}$ by minimizing the following objective:

$$\min_{\mathbf{l}} \|\mathbf{y} - \text{diag}(\mathbf{r})\mathbf{l}\|_2^2 + \mu_g \sum_{k=1}^{N} \mathbf{l}^\top \mathbf{S}_{r,k}^\top \mathcal{L}_{r,k} \mathbf{S}_{r,k} \mathbf{l} + \mu_g \sum_{k=1}^{N} \mathbf{l}^\top \mathbf{S}_{c,k}^\top \mathcal{L}_{c,k} \mathbf{S}_{c,k} \mathbf{l}, \tag{4.42}$$

where μ_g is the tradeoff parameter. Similarly, the solution $\mathbf{l}^*$ to (4.42) can be obtained by solving a linear system:

$$\left(\text{diag}^2(\mathbf{r}) + \mu_g \sum_{k=1}^{N} (\mathbf{S}_{r,k}^\top \mathcal{L}_{r,k} \mathbf{S}_{r,k} + \mathbf{S}_{c,k}^\top \mathcal{L}_{c,k} \mathbf{S}_{c,k}) \right) \mathbf{l}^* = \text{diag}(\mathbf{r})\mathbf{y}. \tag{4.43}$$

$\mathbf{l}^*$ in (4.43) again can be obtained via CG without matrix inversion.

4.4.4 Algorithm Summary

Having obtained $\mathbf{l}^*$, a new solution $\mathbf{r}^*$ to (4.39) can be computed again using the updated $\mathbf{l}^*$, and new graphs with edges updated via (4.37) using the most recently computed $\mathbf{r}$. This iterative update of edge weights means that the GLRs are signal-dependent and thus promote PWC reflectance reconstruction. Having obtained $\mathbf{r}^*$, a new solution $\mathbf{l}^*$ to (4.43) can be sought using recomputed $\mathbf{r}^*$ and new GNG Laplacians

with edges updated via (4.40). The algorithm iterates until convergence. Similarly, this iterative edge weight update means that the GGLRs are signal-dependent and promote PWP illumination reconstruction.

Having obtained solutions $\mathbf{l}^*$ and $\mathbf{r}^*$, we construct a contrast-enhanced pixel $x_{i,j}$ via gamma correction [34] based on (3.14).

4.4.5 Computation Considerations

Due to small illumination values in $\mathbf{l}$, the condition number can be large in the coefficient matrix in (4.39), which decreases the speed of CG computations (Similarly, for linear system (4.43) the condition number can be large due to small reflectance values in $\mathbf{r}$). To improve the computation speed of CG, we perform *preconditioning* [58] as follows.

Generally, in place of linear system $\mathbf{Ax} = \mathbf{b}$, we can consider equivalent $\mathbf{PAx} = \mathbf{Pb}$ instead for invertible matrix $\mathbf{P}$:

$$\begin{aligned} \mathbf{PAP}^\top(\mathbf{P}^\top)^{-1}\mathbf{x} &= \mathbf{Pb} \\ \mathbf{PAP}^\top\hat{\mathbf{x}} &= \hat{\mathbf{b}} \end{aligned} \tag{4.44}$$

where $\mathbf{P}^\top(\mathbf{P}^\top)^{-1} = \mathbf{I}$, $\hat{\mathbf{x}} = (\mathbf{P}^\top)^{-1}\mathbf{x}$, and $\hat{\mathbf{b}} = \mathbf{Pb}$. Note that coefficient matrix $\mathbf{PAP}^\top$ in (4.44) is also symmetric and PD given $\mathbf{A}$ is symmetric and PD by assumption, and thus CG can solve (4.44). If $\kappa(\mathbf{PAP}^\top) \ll \kappa(\mathbf{A})$, then we have improved the conditioning of the linear system, and CG will run faster. The challenge is to design an invertible $\mathbf{P}$ satisfying this condition.

One simple preconditioner that is easily invertible is a diagonal matrix (called *Jacobi* in the linear algebra literature [58]). We propose one variant: $\mathbf{P} = \text{diag}(\mathbf{p})$,

where $p_i = A_{i,i}^{-1/2}$. This means $\mathbf{PAP}^\top = \text{diag}(\mathbf{p})\mathbf{A}\text{diag}(\mathbf{p})$ has ones along its diagonal. Note that $A_{i,i} > 0$ for $\mathbf{A}$ in (4.39), since $\mathbf{l}$ is strictly positive and diagonals of $\mathbf{H}_k^\top \mathbf{L}_{r,k}\mathbf{H}_k$ and $\mathbf{G}_k^\top \mathbf{L}_{c,k}\mathbf{G}_k$ are non-negative. Note also that $\mathbf{A}$ in (4.39) is *diagonally dominant*, *i.e.*, $A_{i,i} > \sum_{j \neq i} |A_{i,j}|$—a matrix condition where Jacobi preconditioner is known to perform well.

Thus, when solving for $\mathbf{r}^*$ in (4.39), we first solve for $\hat{\mathbf{r}}^*$ via linear system $\text{diag}(\mathbf{p})\mathbf{A}\,\text{diag}(\mathbf{p})\,\hat{\mathbf{r}}^* = \text{diag}(\mathbf{p})\,\mathbf{b}$. We then obtain solution $\mathbf{r}^* = \text{diag}(\mathbf{p})\,\hat{\mathbf{r}}^*$. A similar procedure is employed when solving for $\mathbf{l}^*$ in (4.43).

Chapter 5

Experimental Results

5.1 Depth Image Denoising

5.1.1 Experimental Setup

We selected five depth maps from the New Tsukuba stereo dataset [2]. We added zero-mean white Gaussian noise with standard deviations $20, 30$, and 40 to each of the five depth maps to evaluate our performance under different noising levels.

When applying the algorithm, the standard deviation σ of the AWGN can be accurately estimated [62], thus we assume that it is prior knowledge. We use three parameter settings for the parameters μ_{g} and μ_{p}, and the weight parameter corresponding to the standard deviation applied to the depth map.

5.1.2 Experimental Results

We compared our method against seven competing schemes: patch-based GLR [18] and GGLR [33], total variation [63], BF, two non-local methods NLGBT [44] and BM3D [43], and finally, DnCNN [15], which is a convolutional neural network specifically trained for image denoising on natural images.

In [18, 33, 63] and BF, where a specific set of parameters are not introduced, we empirically set the parameters to ensure the methods produce visually pleasing results and use the default setting in other methods. The available pretrained version of the DnCNN is trained on natural images and is targeted for 8-bit images. To adapt this network for depth images, we scaled and shifted the pixel values to be between 0 and 255, and then quantized the intensities to encode the depth map into an 8-bit image compatible with DnCNN. Once the encoded image was denoised using DnCNN, we dequantized it and reversed the scaling process to obtain a denoised version.

In our objective evaluation, we have examined our depth-image denoising method using two metrics: peak signal-to-noise ratio (PSNR) [64] and structural similarity index measure (SSIM) [65]. A higher PSNR indicates that the algorithm has performed well in minimizing the noise and preserving the details when compared to the ground truth image. Moreover, a higher SSIM indicates that the noisy image retains more of the original image's structural information. Tables 5.1 and 5.2 demonstrate the effectiveness of our method in denoising. The highest PSNR and SSIM values are highlighted in bold and coloured blue, while the second highest values are coloured red. Our method outperformed GLR, GGLR, TV, BF, and DnCNN[1] and produced comparable results against non-local methods NLGBT and BM3D while maintaining a linear time complexity. On the other hand, we outperformed all of the methods in terms of SSIM by up to 0.306, except for cases where the noise variance is 40, where NLGBT performed marginally better than our method.

For a qualitative illustration, we have provided the texture image, clean and noisy

[1]We observe that DnCNN effectively denoised images with high noise standard deviations, while maintaining the quality of images with lower noise levels, demonstrating its robustness to noise. However, it did not achieve state-of-the-art performance in depth image denoising, primarily because it was pre-trained on natural images and was optimized for 8-bit images.

depth maps, and denoised outputs of competing methods in Figure 5.1. It is evident that BF, TV, and GGLR outputs still contain noise, while GLR introduced blocking artifacts. DnCNN tended to oversmooth the image. For a more detailed comparison, Figure 5.2 provides a closer examination of the results from BM3D, NLGBT, and our method. We have used a colour map with a larger dynamic range[2] and rescaled the display range of the enlarged image segments based on the depth values for better illustration. It is apparent that although all methods significantly reduced noise, our method excelled in preserving image details, whereas BM3D and NLGBT eliminated object details.

Table 5.1: Denoising quality comparison in PSNR (dB) of different methods.

Test Image	Noise std (σ)	PSNR								
		noisy image	GLR	GGLR	TV	BF	DnCNN	NLGBT	BM3D	ours
R00434	20	33.20	40.65	40.26	45.11	44.19	39.84	**49.19**	48.11	47.58
	30	29.68	37.75	37.49	41.68	40.54	39.75	**46.06**	45.42	45.88
	40	27.19	33.01	33.97	40.91	38.03	39.53	40.85	43.52	**44.02**
R01709	20	31.62	42.88	43.53	45.83	43.82	43.03	49.21	**49.37**	48.96
	30	28.10	37.65	38.65	41.29	40.43	42.80	46.10	46.78	**47.14**
	40	25.60	32.01	33.37	40.58	37.92	42.60	**45.07**	44.97	45.05
R01610	20	30.36	40.38	40.38	43.77	42.10	40.57	**46.99**	46.93	46.00
	30	26.83	36.17	36.13	39.65	38.77	40.16	43.92	44.28	**44.39**
	40	24.34	30.59	31.81	38.87	36.31	39.94	41.49	42.42	**42.54**
R01350	20	36.34	45.10	44.26	47.93	47.72	40.77	**52.44**	51.53	47.10
	30	32.83	41.46	41.20	44.71	44.11	40.71	**48.90**	48.80	46.43
	40	30.35	36.49	37.37	43.99	41.59	40.63	-	**46.89**	45.57
R01120	20	28.54	41.84	41.97	43.71	41.13	44.06	46.64	47.79	**49.37**
	30	25.02	35.32	35.49	38.56	37.55	43.09	43.68	45.24	**46.22**
	40	22.52	28.97	30.33	37.85	35.00	42.35	43.33	**43.41**	43.16

[2]Ratio between the brightest and darkest pixels of the image.

Table 5.2: Denoising quality comparison in SSIM of different methods.

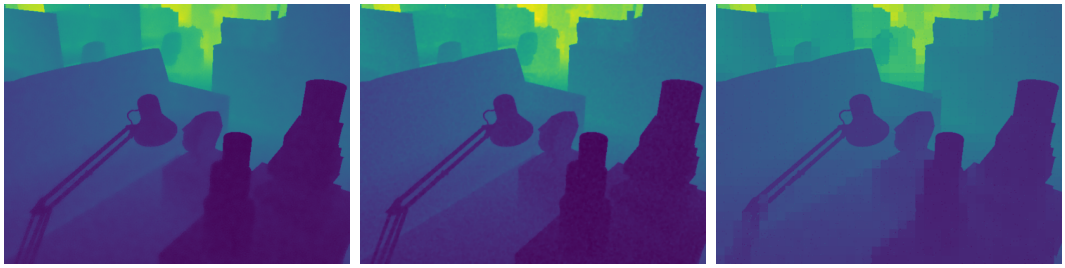
Test Image	Noise std (σ)	SSIM								
		noisy image	GLR	GGLR	TV	BF	DnCNN	NLGBT	BM3D	ours
R00434	20	0.114	0.226	0.373	0.342	0.220	0.289	0.517	0.502	**0.772**
	30	0.091	0.161	0.225	0.205	0.168	0.285	0.403	0.417	**0.654**
	40	0.076	0.101	0.124	0.202	0.141	0.276	**0.570**	0.362	0.522
R01709	20	0.043	0.162	0.349	0.278	0.147	0.267	0.455	0.448	**0.745**
	30	0.031	0.097	0.180	0.130	0.097	0.257	0.334	0.355	**0.617**
	40	0.025	0.042	0.072	0.129	0.073	0.251	**0.476**	0.296	0.472
R01610	20	0.068	0.179	0.327	0.279	0.161	0.292	0.443	0.449	**0.732**
	30	0.052	0.117	0.188	0.150	0.117	0.276	0.327	0.359	**0.595**
	40	0.043	0.062	0.095	0.147	0.094	0.268	0.451	0.303	**0.453**
R01350	20	0.106	0.228	0.329	0.278	0.183	0.163	0.459	0.463	**0.737**
	30	0.086	0.158	0.197	0.170	0.143	0.162	0.351	0.377	**0.594**
	40	0.073	0.099	0.114	0.169	0.121	0.160	**0.473**	0.319	0.453
R01120	20	0.032	0.152	0.306	0.221	0.116	0.333	0.380	0.417	**0.723**
	30	0.023	0.085	0.153	0.101	0.077	0.293	0.272	0.325	**0.543**
	40	0.019	0.034	0.057	0.100	0.058	0.265	**0.419**	0.264	0.379

5.2 Low Light Image Contrast Enhancement and Denoising

5.2.1 Experimental Setup

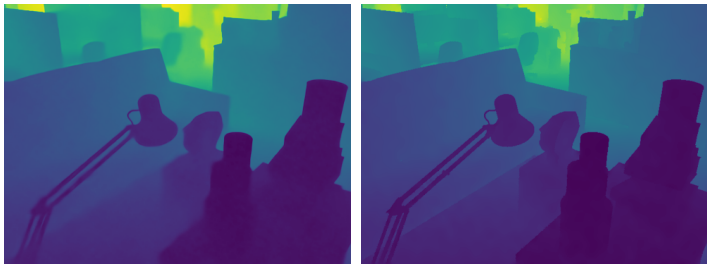
We conducted experiments using MATLAB R2022b on an Apple M2 chip with 8GB RAM to evaluate the performance of our proposed method. We selected twelve images from the datasets provided in [10] and [49]. To ensure compatibility with the chosen patch size, we adjusted the size of each image so that the height and width were divisible by 5. We added zero-mean Gaussian noise with a standard deviation of 0.001 to every image pixel. Four parameters in our method, μ_r in (4.38), σ_r in (4.37), μ_l in (4.42), and σ_l in (4.40), were empirically set to 1, 1, 0.1, and 0.2, respectively. More generally, weight parameters μ_r and μ_l can be chosen to minimize mean squared error [66]. We set the convergence tolerance for CG to $\epsilon = 10^{-6}$.

(a) Texture image (b) Ground truth depth map (c) Noisy depth map ($\sigma = 30$)



(d) BF (e) TV (f) GLR

(g) GGLR (h) BM3D (i) NLGBT

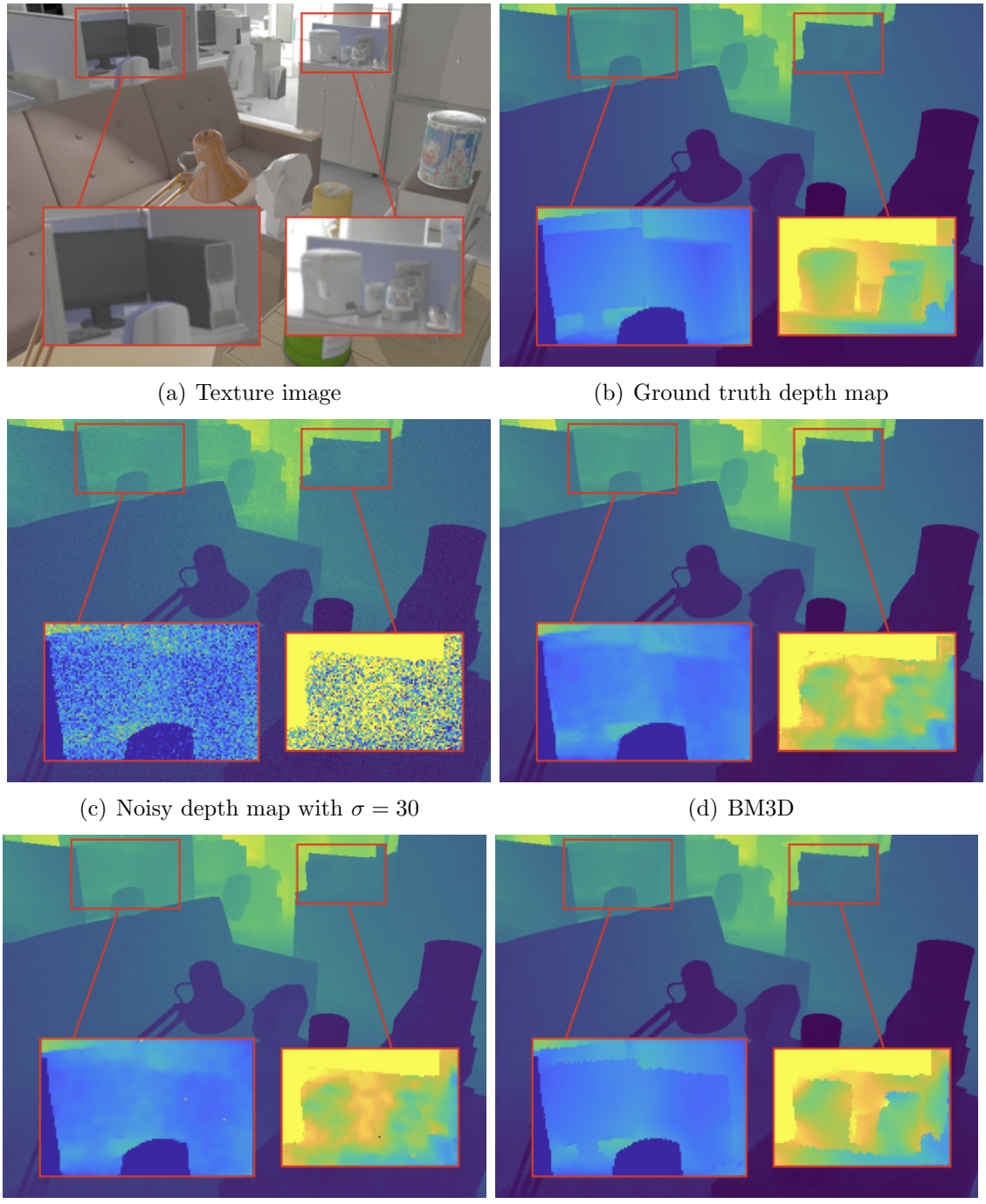


(j) DnCNN (k) Proposed

Figure 5.1: Visual comparison of different methods on `R00434` selected from New Tsukuba Stereo Dataset [2].

5.2.2 Experimental Results

We compared our method against four competing schemes [10, 49–51]. The first two methods focused on contrast enhancement, while the latter two employed joint



(a) Texture image

(b) Ground truth depth map

(c) Noisy depth map with $\sigma = 30$

(d) BM3D

(e) NLGBT

(f) ours

Figure 5.2: Close-up visual comparison of different methods on `R00434`.

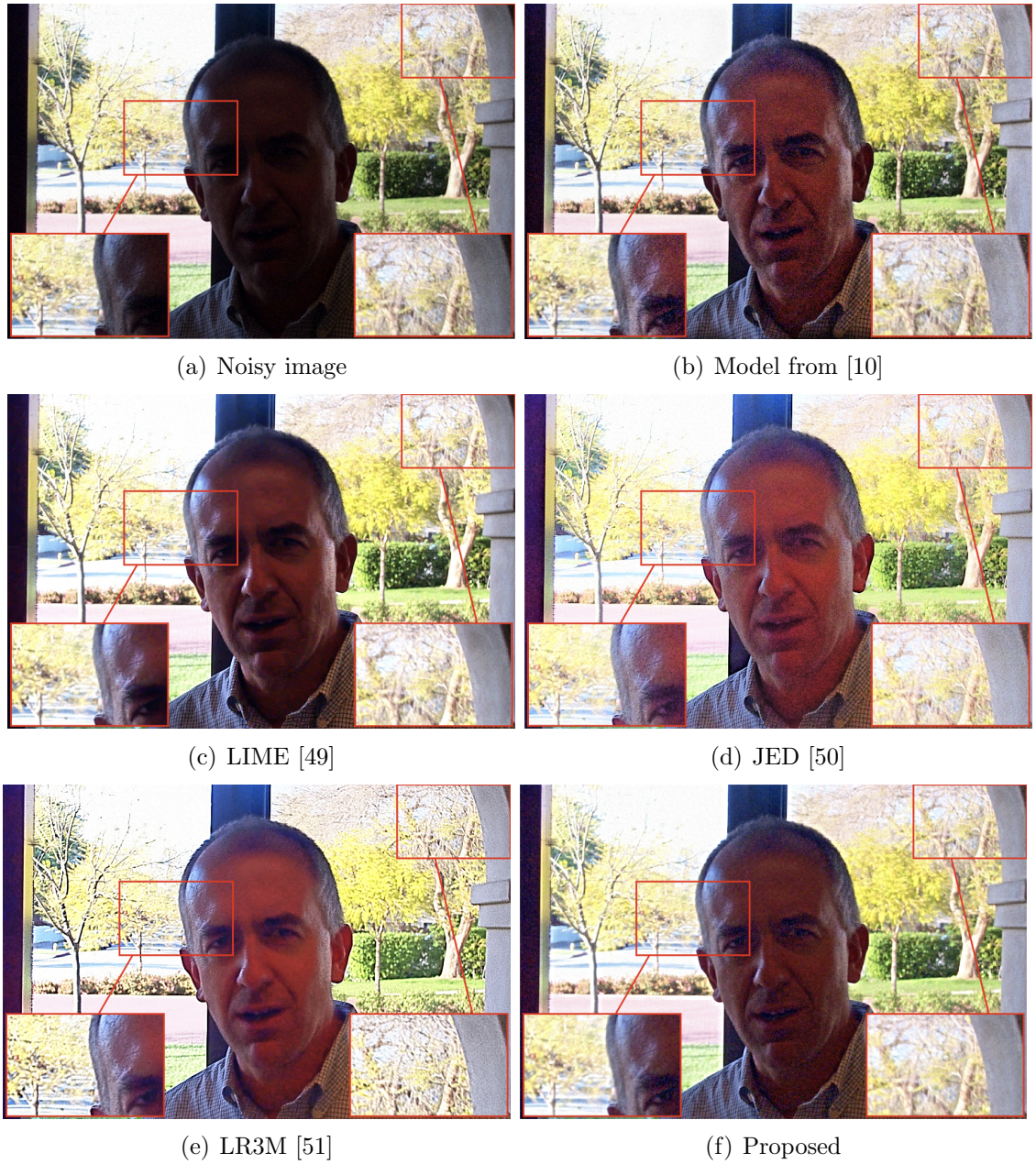
denoising and contrast enhancement. Note that [49] incorporated BM3D [43] as a post-denoising step, where we chose $\sigma = 10$. Towards a fair comparison, we adjusted the brightness parameter in all five methods so that they produced images with roughly the same brightness level.

Figures 5.3 and 5.4 show visual comparisons of different methods. The first and third schemes noticeably amplified the noise when performing contrast enhancement, resulting in noisier outputs compared to our method. Meanwhile, the second method employed denoising as a post-processing technique, resulting in blurring of image details. In comparison, our method produced results that are comparable in quality to the LR3M model with a significantly faster computation speed as demonstrated in Table 5.4. Further, as shown in Figure 5.4, LR3M can lead to image over-smoothing.

In our objective evaluation, we have examined our method in contrast enhancement using two metrics: lightness-order-error (LOE) and Minkowski distance based metric (MDM). Table 5.3 demonstrates the effectiveness of our method in achieving a visually pleasing and realistic output. We outperformed other techniques in terms of LOE [67], a no-reference image quality metric to assess naturalness in enhanced images. LOE specifically assesses the preservation of lightness order by comparing the enhanced image with the original image, without requiring any additional reference images. A lower LOE score indicates a higher level of preservation of lightness order and, consequently, a more visually pleasing and realistic output. Additionally, we achieved comparable results in MDM, another no-reference quality assessment metric where higher scores indicate higher image quality.

We observe that our preconditioner can greatly reduce the condition number of the coefficient matrix in (4.39) and (4.43). Specifically, in some cases, our preconditioner

reduces very large condition numbers, which exceeded 5200, by up to 80%. These findings suggest that our preconditioners can improve the speed of CG execution.

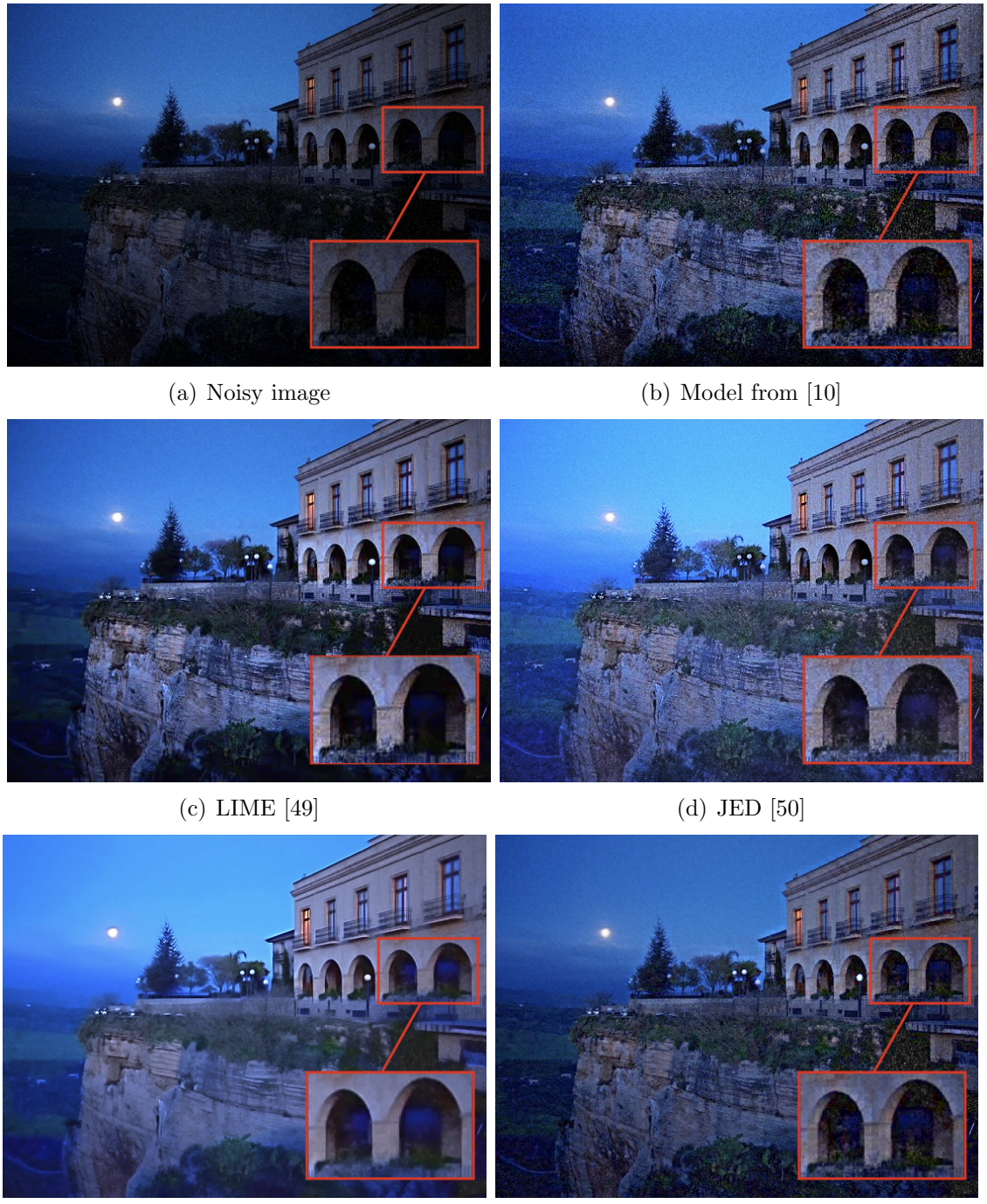


(a) Noisy image (b) Model from [10]

(c) LIME [49] (d) JED [50]

(e) LR3M [51] (f) Proposed

Figure 5.3: Visual comparison of different methods on `Man`.



(a) Noisy image (b) Model from [10]

(c) LIME [49] (d) JED [50]

(e) LR3M [51] (f) Proposed

Figure 5.4: Visual comparison of different methods on `Moonlight`.

Table 5.3: Contrast enhancement quality comparison of different methods based on LOE and MDM measures.

Image/Method	LOE [67]					MDM [68]					
	M1 [10]	LIME [49]	JED [50]	LR3M [51]	ours	Noisy input	M1 [10]	LIME [49]	JED [50]	LR3M [51]	ours
Moonlight	0.6185	0.5444	1.0730	0.9590	**0.4817**	0.9594	0.9522	**0.9683**	0.9652	0.9672	0.9663
Cars	0.5887	0.7051	1.0487	1.0257	**0.5479**	**0.9611**	0.9440	0.9553	0.9483	0.9514	0.9578
Plant	0.8414	0.7030	**0.6998**	0.8754	0.8006	0.9633	0.9582	**0.9748**	0.9709	0.9739	0.9726
Bear	0.8517	**0.6703**	0.9291	1.3507	0.7682	0.8384	0.9076	**0.9760**	0.9750	0.9756	0.9716
Nightfall	0.8879	0.7364	0.4850	0.7349	**0.4701**	0.9711	0.9643	0.9763	0.9735	**0.9767**	0.9741
Street	0.6415	**0.5858**	0.7661	1.2604	0.7317	0.9706	0.9682	0.9762	**0.9785**	0.9767	0.9754
Stormtrooper	0.9141	0.7486	1.2178	0.8096	**0.6590**	0.9689	0.9612	0.9745	0.9716	0.9752	**0.9760**
Riverside	0.9684	0.6977	0.7341	1.1203	**0.5003**	0.9707	0.9589	**0.9758**	0.9721	0.9707	0.9739
Landmark	0.8279	0.6800	1.0666	0.8346	**0.5080**	0.9571	0.9435	**0.9613**	0.9465	0.9515	0.9601
Man	0.6675	0.8141	0.8456	1.2376	**0.5209**	**0.9372**	0.9243	0.9176	0.9107	0.9048	0.9224
Lamp	0.6135	**0.5644**	1.1641	1.9776	0.7960	0.9702	0.9612	**0.9733**	0.9721	0.9709	0.9705
Wire	0.6875	**0.6515**	1.2575	1.4944	0.9911	0.9634	0.9575	**0.9782**	0.9770	**0.9782**	0.9729
Average	0.7591	0.6751	0.9406	1.1400	**0.6480**	0.9526	0.9501	**0.9673**	0.9635	0.9644	0.9661

Table 5.4: Computation complexity comparison of LR3M [51] and our method based on running time in seconds.

Image	Resolution	LR3M [51]	proposed
Moonlight	560×420	250.48	**25.17**
Cars	370×415	284.00	**16.24**
Plant	500×375	142.08	**21.91**
Bear	490×365	162.24	**20.82**
Nightfall	690×460	301.75	**35.20**
Street	1035×785	1442.23	**88.24**
Stormtrooper	450×450	87.36	**21.09**
Riverside	720×680	366.41	**52.82**
Landmark	540×720	148.01	**42.59**
Man	895×590	246.40	**56.47**
Lamp	450×500	165.40	**24.37**
Wire	325×325	152.29	**11.92**
	Average	312.39	**34.74**

Chapter 6

Conclusion

In this thesis, we investigated the restoration of PWP images using graph-based priors focusing on two examples: (i) depth image denoising, and (ii) low-light image contrast enhancement. We formulated convex quadratic optimization objectives with regularization terms specified using graph Laplacian matrices that promote desired characteristics in each scenario. For depth image denoising we employed a GGLR prior to promote PWP signal reconstruction, while simultaneously using a signal-independent GLR prior, with edge weights set to one, to encourage signal continuity. For contrast enhancement, we decomposed the image into reflectance and illumination using Retinex theory [5], and in turn, used a GLR prior to promote a PWC signal for reflectance, and a GGLR prior to promote a PWP signal for illumination.

Our method for depth image denoising outperforms all other baselines in terms of SSIM and produces comparable results in terms of PSNR. Additionally, our signal reconstruction approach enables joint denoising and contrast enhancement, unlike most of the baselines that focus solely on either denoising or contrast enhancement. Among the baselines that address both tasks, we achieve comparable results in terms of MDM while outperforming all the baselines in terms of LOE. Moreover, we not

only deliver competitive results across the tasks we consider, but our approach is significantly faster, running in linear time, demonstrating the practicality of graph-based signal processing priors.

While our signal reconstruction results are promising, it is important to note that all model-based methods tend to struggle in high noise regimes, as evidenced by some of our results in Tables 5.1 and 5.2. Given the increasing prevalence of deep learning models, we believe that integrating our graph-based priors with deep learning methods could potentially help mitigate these shortcomings. A promising direction is *algorithm unrolling* [69], where the hyperparameters of our algorithm (such as μ_{p}, μ_{g}, and ρ) can be optimized to best fit a large training dataset. This hybrid approach could leverage the strengths of both model-based and data-driven methods to achieve superior performance.

Finally, we believe our denoising algorithm can further benefit many downstream applications. For instance, depth maps are often used to generate 3D point clouds, and our method of depth map denoising may significantly enhance the quality of these 3D point clouds, making them more suitable for further processing. Additionally, our joint denoising and contrast enhancement approach can improve various techniques that utilize these images, such as object detection. We leave the exploration of these applications to future work.

Bibliography

[1] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *German Conference on Pattern Recognition*, 2014.

[2] M. P. Martorell, A. Maki, S. Martull, Y. Ohkawa, and K. Fukui, "New tsukuba stereo dataset," 2012. Accessed: 2024-06-24.

[3] M. Antunes, J. P. Barreto, and U. Nunes, "Piecewise-planar reconstruction using two views," *Image and Vision Computing*, vol. 46, pp. 47–63, 2016.

[4] S. N. Sinha, D. Steedly, and R. Szeliski, "Piecewise planar stereo for image-based rendering," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 1881–1888, 2009.

[5] E. Land, "The Retinex theory of color vision," in *Scientific American*, vol. 61, no.1, pp. 1–11, 1977.

[6] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[7] C. Vogel and M. Oman, "Fast, robust total variation-based reconstruction of noisy, blurred images," *IEEE Transactions on Image Processing*, vol. 7, no. 6, pp. 813–824, 1998.

[8] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2702–2719, 2020.

[9] R. Mekuria, K. Blom, and P. César, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 4, pp. 828–842, 2017.

[10] X. Fu, D. Zeng, Y. Huang, X. Zhang, and X. Ding, "A weighted variational model for simultaneous reflectance and illumination estimation," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2782–2790, IEEE Computer Society, 2016.

[11] C. Wei, W. Wang, W. Yang, and J. Liu, "Deep Retinex decomposition for low-light enhancement," arXiv:1808.04560, 2018.

[12] C. Li, C. Guo, L. Han, J. Jiang, M.-M. Cheng, J. Gu, and C. C. Loy, "Low-light image and video enhancement using deep learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 9396–9416, 2022.

[13] K. G. Lore, A. Akintayo, and S. Sarkar, "Llnet: A deep autoencoder approach to natural low-light image enhancement," *Pattern Recognition*, vol. 61, pp. 650–662, 2017.

[14] J. Yang, X. Jiang, C. Pan, and C.-L. Liu, "Enhancement of low light level images with coupled dictionary learning," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 751–756, 2016.

[15] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *Trans. Img. Proc.*, vol. 26, p. 3142–3155, jul 2017.

[16] S. M. Valiollahzadeh, H. Firouzi, M. Babaie-Zadeh, and C. Jutten, "Image denoising using sparse representations," in *Independent Component Analysis and Signal Separation, 8th International Conference, ICA 2009, Paraty, Brazil, March 15-18, 2009. Proceedings* (T. Adali, C. Jutten, J. M. T. Romano, and A. K. Barros, eds.), vol. 5441 of *Lecture Notes in Computer Science*, pp. 557–564, Springer, 2009.

[17] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[18] J. Pang and G. Cheung, "Graph Laplacian regularization for inverse imaging: Analysis in the continuous domain," in *IEEE Transactions on Image Processing*, vol. 26, no.4, pp. 1770–1785, April 2017.

[19] X. Zhang, G. Cheung, J. Pang, Y. Sanghvi, A. Gnanasambandam, and S. H. Chan, "Graph-based depth denoising dequantization for point cloud enhancement," *IEEE Transactions on Image Processing*, vol. 31, pp. 6863–6878, 2022.

[20] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," in *Proceedings of the IEEE*, vol. 106, no.5, pp. 808–828, May 2018.

[21] G. Cheung, E. Magli, Y. Tanaka, and M. Ng, "Graph spectral image processing," in *Proceedings of the IEEE*, vol. 106, no.5, pp. 907–930, May 2018.

[22] K. Bredies and D. A. Lorenz, "Linear convergence of iterative soft-thresholding," *Journal of Fourier Analysis and Applications*, vol. 14, pp. 813–837, 2007.

[23] N. Parikh and S. Boyd, "Proximal algorithms," in *Foundations and Trends in Optimization*, vol. 1, no.3, pp. 123–231, 2013.

[24] F. Chen, G. Cheung, and X. Zhang, "Manifold graph signal restoration using gradient graph laplacian regularizer," *arXiv preprint arXiv:2206.04245*, 2022.

[25] W. Hu, G. Cheung, A. Ortega, and O. Au, "Multi-resolution graph Fourier transform for compression of piecewise smooth images," in *IEEE Transactions on Image Processing*, vol. 24, no.1, pp. 419–433, January 2015.

[26] J. Pang, G. Cheung, A. Ortega, and O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, (Brisbane, Australia), April 2015.

[27] Y. Bai, G. Cheung, X. Liu, and W. Gao, "Graph-based blind image deblurring from a single photograph," *IEEE Transactions on Image Processing*, vol. 28, no.3, pp. 1404–1418, 2019.

[28] X. Liu, G. Cheung, X. Ji, D. Zhao, and W. Gao, "Graph-based joint dequantization and contrast enhancement of poorly lit JPEG images," *IEEE Transactions on Image Processing*, vol. 28, no.3, pp. 1205–1219, March 2019.

[29] C. Dinesh, G. Cheung, and I. V. Bajić, "Point cloud denoising via feature graph Laplacian regularization," *IEEE Transactions on Image Processing*, vol. 29, pp. 4143–4158, 2020.

[30] X. Liu, G. Cheung, X. Wu, and D. Zhao, "Random walk graph Laplacian based smoothness prior for soft decoding of JPEG images," *IEEE Transactions on Image Processing*, vol. 26, no.2, pp. 509–524, February 2017.

[31] J. Zeng, G. Cheung, M. Ng, J. Pang, and C. Yang, "3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model," *IEEE Transactions on Image Processing*, vol. 29, pp. 3474–3489, 2020.

[32] C. Dinesh, G. Cheung, and I. V. Bajić, "Point cloud video super-resolution via partial point coupling and graph smoothness," *IEEE Transactions on Image Processing*, vol. 31, pp. 4117–4132, 2022.

[33] F. Chen, G. Cheung, and X. Zhang, "Manifold graph signal restoration using gradient graph laplacian regularizer," *IEEE Transactions on Signal Processing*, pp. 1–16, 2024.

[34] R. C. Gonzalez and R. E. Woods, *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008.

[35] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 839–846, 1998.

[36] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar conference on signals, systems and computers*, pp. 40–44, IEEE, 1993.

[37] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[38] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[39] A. Chambolle and P.-L. Lions, "Image recovery via total variation minimization and related problems," *Numerische Mathematik*, vol. 76, pp. 167–188, 1997.

[40] F. Chen, G. Cheung, and X. Zhang, "Fast & robust image interpolation using gradient graph laplacian regularizer," in *2021 IEEE International Conference on Image Processing, ICIP 2021, Anchorage, AK, USA, September 19-22, 2021*, pp. 1964–1968, IEEE, 2021.

[41] K. Bredies, K. Kunisch, and T. Pock, "Total generalized variation," *SIAM Journal on Imaging Sciences*, vol. 3, no. 3, pp. 492–526, 2010.

[42] J. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," tech. rep., USA, 1994.

[43] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising with block-matching and 3d filtering," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 6064, pp. 354–365, 02 2006.

[44] W. Hu, X. Li, G. Cheung, and O. Au, "Depth map denoising using graph-based transform and group sparsity," in *2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 001–006, 2013.

[45] G. Shen, W.-S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *28th Picture Coding Symposium*, pp. 566–569, 2010.

[46] T. Goldstein and S. Osher, "The split bregman method for l1-regularized problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[47] R. Kimmel, M. Elad, D. Shaked, R. Keshet, and I. Sobel, "A variational framework for Retinex," *Int. J. Comput. Vision*, vol. 52, p. 7–23, apr 2003.

[48] W. Ma, J.-M. Morel, S. Osher, and A. Chien, "An l1-based variational model for Retinex theory and its application to medical images," in *CVPR 2011*, pp. 153–160, 2011.

[49] X. Guo, Y. Li, and H. Ling, "LIME: low-light image enhancement via illumination map estimation," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 982–993, 2017.

[50] X. Ren, M. Li, W. Cheng, and J. Liu, "Joint enhancement and denoising method via sequential decomposition," *CoRR*, vol. abs/1804.08468, 2018.

[51] X. Ren, W. Yang, W.-H. Cheng, and J. Liu, "LR3M: Robust low-light enhancement via low-rank regularized Retinex model," *IEEE Transactions on Image Processing*, vol. 29, pp. 5862–5876, 2020.

[52] L. Yao and Z. Pan, "The Retinex-based image dehazing using a particle swarm optimization method," *Multimedia Tools and Applications*, vol. 80, p. 3425–3442, 2021.

[53] A. Lecert, R. Fraisse, A. Roumy, and C. Guillemot, "A new regularization for Retinex decomposition of low-light images," in *2022 IEEE International Conference on Image Processing (ICIP)*, pp. 906–910, 2022.

[54] T. Goldstein and S. Osher, "The split bregman method for l1-regularized problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[55] Z. Lin, R. Liu, and Z. Su, "Linearized alternating direction method with adaptive penalty for low-rank representation," in *Advances in Neural Information Processing Systems* (J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, eds.), vol. 24, Curran Associates, Inc., 2011.

[56] W. Dong, G. Shi, and X. Li, "Nonlocal image restoration with bilateral variance estimation: A low-rank approach," *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 700–711, 2013.

[57] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*, (Berkeley and Los Angeles), pp. 481–492, University of California Press, 1951.

[58] G. Golub and C. F. V. Loan, *Matrix Computations (Johns Hopkins Studies in the Mathematical Sciences)*. Johns Hopkins University Press, 2012.

[59] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 01 2011.

[60] J. Sherman and W. J. Morrison, "Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix," *The Annals of Mathematical Statistics*, vol. 21, no. 1, pp. 124 – 127, 1950.

[61] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[62] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[63] A. Chambolle, V. Caselles, M. Novaga, D. Cremers, and T. Pock, "An introduction to total variation for image analysis," vol. 9, 01 2010.

[64] Q. M. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.

[65] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[66] P. Chen and S. Liu, "Bias-variance tradeoff of graph laplacian regularizer," *IEEE Signal Process. Lett.*, vol. 24, no. 8, pp. 1118–1122, 2017.

[67] S. Wang, J. Zheng, H. Hu, and B. Li, "Naturalness preserved enhancement algorithm for non-uniform illumination images," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3538–3548, 2013.

[68] H. Z. Nafchi and M. Cheriet, "Efficient no-reference quality assessment and classification model for contrast distorted images," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 518–523, 2018.

[69] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.