Graph-based regularization of inverse problems in imaging

Présentée le 29 mai 2020

t à la Faculté des sciences et techniques de l’ingénieur
Laboratoire de traitement des signaux 4
Programme doctoral en génie électrique

pour l’obtention du grade de Docteur ̄es Sciences

par

Mattia ROSSI

Acceptée sur proposition du jury

Prof. J.-Ph. Thiran, président du jury
Prof. P. Frossard, directeur de thèse
Dr R. Timofte, rapporteur
Dr S. Galliani, rapporteur
Dr M. Salzmann, rapporteur
To my parents, Giorgio and Patrizia.
Acknowledgements

First I would like to thank my PhD advisor, Prof. Pascal Frossard, who made my PhD journey possible. He believed in my research potential, he was patient and he provided his scientific and moral support throughout my time at the LTS4 Lab, especially in those moments when my research seemed stuck. I am very grateful to him.

I would like to extend my sincerest thanks to the jury members of my thesis committee, Dr. Mathieu Salzmann, Dr. Radu Timofte, Dr. Silvano Galliani and Prof. Jean-Philippe Thiran, for reviewing this manuscript. I appreciated their comments on the manuscript and the fruitful discussion during the exam session.

I want to thank also Prof. Giancarlo Calvagno who introduced me to Image Processing during my MSc. thesis at the University of Padua and encouraged me to pursue a PhD abroad.

I am very grateful to Mireille, whom I luckily met in the middle of my PhD journey. She is a very knowledgeable researcher and a very nice friend. She supported me not only with fruitful scientific discussions but especially with her very positive character and mood.

An internship at Sony in Stuttgart, Germany, offered me the opportunity to work with great researchers at the Computational Imaging Group and allowed me to broaden my research expertise. I would like to thank Mr. Oliver Erdler, Dr. Andreas Kuhn and Dr. Francesco Michielin for making the internship possible.

Many thanks to the former and current LTS4 labmates. In particular, I would like to thank Francesca and Roberto, for their continuous attempts to make me always see the positive side of things. My special thanks go to Stefano and Seyed, whose PhD journeys at LTS4 overlapped with mine. I met Stefano on my day one in Switzerland: we enjoyed any kind of discussion, scientific and non, afternoon breaks, laughs. I miss them. Seyed has been my officemate for most of my PhD time: he is a brilliant researcher and I am glad to have had the chance to discuss and share thoughts with him.

I am also happy to have met Niccolò, my flatmate during my time in Lausanne. I want to thank him for our meals together, for our trips from Switzerland to Italy and vice versa, for our late evening scientific discussions and in general for all the time we enjoyed together. I believe I found another nice friend.

Last but not the least, I am indebted to my family for their unconditional love, support and encouragement in all my decisions. I am really grateful to them. Having them standing by me makes me feel an extremely fortunate person.

Lausanne, 12 February 2020

Mattia Rossi
Abstract

As of today, the extension of the human visual capabilities to machines remains both a cornerstone and an open challenge in the attempt to develop intelligent systems. On the one hand, the development of more and more sophisticated imaging devices, capable of sensing richer information than a plain perspective projection of the real world, is critical in order to allow machines to understand the complex environment around them. On the other hand, despite the advances in imaging, the complexity of the real world cannot be fully captured by a single imaging device, either due to intrinsic hardware limitations or to the environment complexity itself. As a consequence, the attempt to extend the human visual capabilities to machines requires inevitably to estimate some unknown quantities, which could not be measured, from the available captured data. Equivalently, imaging requires the solution of arbitrarily complex inverse problems.

In most scenarios, inverse problems are ill-posed and admit an infinite number of solutions, while only one or few of them are the desired ones. It becomes therefore crucial to reduce, equivalently to regularize, the solution space by exploiting all the available prior information about the problem structure and, especially, about the target quantity to estimate. In this thesis we investigate the use of graph-based regularizers to encode our prior knowledge about the target quantity and to inject it directly into the inverse problem. In particular, we cast the inverse problem into an optimization task, where the target quantity is modelled as a graph whose topology captures our prior knowledge. In order to show the effectiveness and the flexibility of graph-based regularizers, we study their use in different inverse imaging problems, each one characterized by different geometrical constraints.

We start by investigating how to augment the resolution of a light field. In fact, although light field cameras permit to capture the 3D information in a scene within a single exposure, thus providing much richer information than a perspective camera, their compact design limits their spatial resolution dramatically. We present a smooth graph-based regularizer which models the geometry of a light field explicitly and we use it to augment the light field spatial resolution while relying only on the complementary information encoded in the low resolution light field views. In particular, we show that the use of a graph-based regularizer permits to enforce the light field geometric structure without the need for a precise and costly disparity estimation step.

Then we analyze the further benefits provided by adopting nonsmooth graph-based regularizers, as these better preserve edges and fine details than their smooth counterpart. In particular, we focus on a specific nonsmooth graph-based regularizer and show its effectiveness within
two applications. The first application revolves again around light field super-resolution, which permits a comparison with the smooth regularizer adopted previously. The second applications is disparity estimation in omnidirectional stereo systems, where the two captured images and the target disparity map live on a spherical surface, hence a graph-based regularizer can be used to model the non trivial correlation underneath each signal.

Finally, we investigate the refinement of a depth map and the estimation of the corresponding normal map. In fact, both active depth sensing devices and stereo methods produce depth maps characterized by artifact reflecting their hardware or software limitations, respectively. As a consequence, a later depth refinement step is typically required in order to remove noisy values or estimate missing depth map areas. We present a nonsmooth graph-based regularizer which leverages the planar bias characterizing most human made environments and models the 3D world as piece-wise planar. The proposed graph-based regularizer permits to estimate the scene normals jointly with the refined depth map: this is particularly useful in the context of 3D reconstruction and in scene understanding.

We carry out extensive tests comparing our graph-based approaches to state-of-the-art methods for each considered problem, we analyze graph-based regularization benefits, as well as its possible limitations.
**Résumé**

À ce jour, le transfert des capacités visuelles de l’homme aux machines reste un défi majeur pour la conception de systèmes intelligents. D’une part, le développement des appareils d’imagerie de plus en plus sophistiqués et capables de capter des informations plus riches qu’une simple projection en perspective du monde réel est essentiel pour pouvoir permettre aux machines de comprendre l’environnement compliqué qui les entoure. D’autre part, malgré les progrès dans le domaine du traitement d’image, la complexité du monde réel ne peut pas être entièrement saisie par un seul dispositif d’imagerie, soit en raison de limitations matérielles intrinsèques, soit en raison de la complexité de l’environnement lui-même. En conséquence, la tentative d’étendre les capacités visuelles de l’homme aux machines nécessite inévitablement d’estimer certaines quantités indéterminées, qui ne pourraient pas être mesurées à partir des données disponibles. En équivalence, le traitement d’image nécessite la résolution des problèmes inverses arbitrairement complexes.

Dans la plupart des scénarios, les problèmes inverses sont mal posés et admettent une infinité de solutions dont une seule ou peu d’entre elles sont celles souhaitées. Il devient donc essentiel de réduire, ou en d’autres termes régulariser, l’espace des solutions en exploitant toutes les informations connues au préalable sur la structure du problème et surtout sur la variable cible à estimer. Dans cette thèse, nous étudions les termes de régularisation basés sur des graphes pour coder nos informations connues a priori sur la variable cible et les injecter ensuite dans un problème inverse. En particulier, nous formulons le problème inverse sous forme d’un problème d’optimisation où la variable cible est modélisée comme un graphe dont la topologie reflète nos informations a priori. Afin de montrer l’efficacité et la flexibilité des régularisations basées sur les graphes, nous étudions leur efficacité dans différents types de problèmes inverses dans le domaine du traitement d’image, chacun caractérisé par des régularisations géométriques différentes.

Nous étudions tout d’abord le problème de la super-resolution des images plénoptiques. Bien que les appareils photographiques plénoptiques permettent de capturer les informations 3D d’une scène en une seule prise de vue, fournissant ainsi des informations beaucoup plus riches qu’une seul caméra, leur conception compacte limite considérablement leur résolution spatiale. Nous présentons une régularisation graphique lisse qui modélise explicitement la géométrie d’un champ lumineux et nous l’utilisons pour augmenter la résolution spatiale des images plénoptiques tout en ne s’appuyant que sur les informations complémentaires encodées dans les vues à basse résolution. En particulier, nous montrons que l’utilisation d’une régularisation graphique permet de renforcer la structure géométrique des images sans
avoir besoin d’estimer la disparité.
Nous analysons ensuite les autres avantages offerts par l’adoption de la régularisation graphiques non lisses, car ceux-ci préservent mieux les détails fins dans les images par rapport aux régularisations lisses. Nous nous concentrons en particulier sur une régularisation graphique non lisse spécifique et nous montrons son intérêt dans deux applications. La première application tourne à nouveau autour de la super-résolution des images plénoptiques, qui permet une comparaison avec le régularisateur lisse adopté précédemment. La seconde application concerne l’estimation de la disparité dans les systèmes stéréo omnidirectionnels, où les deux images capturées et la carte de disparité de la cible vivent sur une surface sphérique, d’où l’utilisation d’un régularisateur graphique pour modéliser la corrélation non triviale dans chaque signal.
Enfin, nous étudions le problème de peaufinage d’une carte de profondeur et de l’estimation de la carte des normales associées. En fait, les dispositifs de détection de la profondeur et les méthodes stéréo produisent des cartes de profondeur caractérisées par des artefacts reflétant leurs limitations matérielles ou logicielles. En conséquence, une étape ultérieure de peaufinage de la profondeur est généralement nécessaire pour supprimer les valeurs bruitées ou estimer les zones incomplètes de la carte de profondeur. Nous présentons une régularisation graphique non lisse qui exploite le biais planaire caractérisant la plupart des environnements créés par l’homme et modélise le monde 3D comme un ensemble de plans. La régularisation graphique proposée permet d’estimer les normales de la scène conjointement avec la carte de profondeur peaufinée : ceci est particulièrement utile dans le contexte de la reconstruction 3D et dans la compréhension de la scène.
Nous avons effectué des tests approfondis en comparant nos approches graphiques aux méthodes existantes dans l’état de l’art pour chacun des problèmes considérés, nous avons analysé les avantages de la régularisation graphique ainsi que ses éventuelles limites.
Contents

Acknowledgements i
Abstract iii

1 Introduction 1
  1.1 Brief regularization review ............................... 2
  1.2 Graph-based regularization ............................ 3
  1.3 Thesis outline ........................................ 5

2 Geometrically consistent light field spatial super-resolution 7
  2.1 Related work ........................................... 9
  2.2 Light field structure .................................. 11
  2.3 Problem formulation .................................. 13
  2.4 Super-resolution algorithm ....................... 16
    2.4.1 Warping matrix construction .................. 16
    2.4.2 Regularization graph construction .............. 19
    2.4.3 Optimization algorithm ......................... 21
  2.5 Experiments .......................................... 23
    2.5.1 Experimental settings .......................... 23
    2.5.2 HCI and Stanford light field datasets ............ 25
    2.5.3 Lytro ILLUM dataset ........................ 32
  2.6 Conclusions .......................................... 33

3 Nonsmooth super-resolution and omnidirectional stereo 35
  3.1 Non Local Total Variation ............................ 35
  3.2 Nonsmooth light field super-resolution ............. 37
    3.2.1 Super-resolution algorithm .................. 37
    3.2.2 Experiments ...................................... 38
  3.3 Omnidirectional stereo ................................ 44
    3.3.1 Problem formulation ............................ 44
    3.3.2 Disparity estimation algorithm .................. 47
    3.3.3 Experiments ...................................... 48
  3.4 Conclusions .......................................... 49

vii
In the attempt to create intelligent systems automatizing and possibly extending the human capabilities, imaging definitely plays a key role. In fact, the development of more and more sophisticated imaging devices, no longer limited to the acquisition of plain images, but capable of sensing richer information like the world geometry, permits to model the real world more accurately. However, both intrinsic hardware limitations and the complexity of the real world itself do not always permit to capture the desired information. As an example, a Time-Of-Flight (ToF) sensor on the top of a robot may capture a depth map characterized by noisy or missing areas, which would not permit to correctly model the geometry of the scene and could prevent the robot from interacting successfully with the surrounding environment. In this scenario, the desired but not directly observable signal, i.e., the clean and complete depth map, must be inferred from the observed signal, i.e., the noisy and possibly incomplete depth map.

The relation between the observed signal $v \in \mathbb{R}^M$ and the target signal $u \in \mathbb{R}^N$ can be modeled as follows:

$$v = \mathcal{A}(u) + n$$  \hspace{1cm} (1.1)

where $\mathcal{A}$ is an operator, not necessarily linear, from $\mathbb{R}^N$ to $\mathbb{R}^M$ and $n \in \mathbb{R}^M$ represents an unknown noise signal with a given distribution. In most real world applications $\mathcal{A}$ models a physical process, while $n$ captures either some physical noise, e.g., the solar interference in the ToF example, or the approximations introduced when modelling the physical process with $\mathcal{A}$. Inverse imaging problems are concerned with the recovery of the target signal $u$ from the observed signal $v$. However, in most real world applications, the problem in Eq. (1.1) is ill posed and an infinite number of solutions exists\(^1\). As a consequence, the target signal $u$ is typically estimated as follows:

$$u^* \in \arg\min_u f(u) + \lambda g(u),$$  \hspace{1cm} (1.2)

where $f(\cdot)$ is referred to as the data term, $g(\cdot)$ is referred to as the regularizer and $\lambda \in \mathbb{R}_{\geq 0}$ balances the two terms. In particular, the data term enforces $u$ to be consistent with the

\(^1\)We do not consider the scenario where no solution exists, as the presence of the random noise $n$ excludes it.
observed data according to the relation in Eq. (1.2) and it typically reads as follows:

\[ f(u) = \| v - \mathcal{A}(u) \|^q_p, \]

where \( \| \cdot \|_p \), with \( p \in \mathbb{R}_{\geq 0} \), denotes an \( \ell_p \)-norm or semi-norm and \( q \in \mathbb{R}_{\geq 0} \). The regularizer, on the other hand, encodes some prior knowledge about the structure of the target signal \( u \) and penalizes those signals \( u \) whose structure deviates from the prior knowledge, thus implicitly narrowing the solution space. The formulation in Eq. (1.2) is quite intuitive, but it can also be formally derived within the Maximum A Posteriori (MAP) framework [1].

In this thesis Eq. (1.2) will represent our formulation of a specific imaging problem, where both the target signal \( u \) and the observed signal \( v \) capture the appearance or the geometry of the 3D real world. The data term \( f(\cdot) \) will establish a physical relation between \( u \) and \( v \), such as the blurring introduced by the lens of a camera. The regularizer \( g(\cdot) \) will capture our prior knowledge about the 3D real world and a graph will be used to encode it. For this reason Section 1.1 provides some significative examples of regularizers adopted in inverse imaging problems, while Section 1.2 introduces the main ideas at the basis of graph-based regularization. Finally, Section 1.3 outlines the thesis structure and our main contributions.

### 1.1 Brief regularization review

In the context of natural and medical image restoration the most popular regularizer is the isotropic Total Variation (TV) [2], which models the vectorized image \( u \) as piece-wise constant and it is defined as follows:

\[ TV(u) = \sum_i \| \nabla u(i) \|_2, \tag{1.3} \]

where \( \nabla u(i) \in \mathbb{R}^2 \) denotes the spatial gradient of the image \( u \) at the pixel \( i \in \{1, 2, \ldots, N\} \). The TV bias toward piece-wise constant images becomes clearer when rewriting Eq. (1.3) as follows:

\[ TV(u) = \left\| \| \nabla u(1) \|_2, \ldots, \| \nabla u(i) \|_2, \ldots, \| \nabla u(N) \|_2 \right\|_1, \tag{1.4} \]

In fact, since the \( \ell_1 \)-norm promotes sparse signals, the TV regularizer in Eq. (1.4) promotes images with a sparse gradients \( \nabla u \), hence piece-wise constant images \( u \). Despite its wide usage, modelling the target image as piece-wise constant represents a quite strong assumption. Therefore, in [3], Bredies et al. introduced the Total Generalized Variation (TGV) regularizer, which permits to relax the piece-wise constant constraint towards a more general and flexible piece-wise planar one.

In some scenarios, the recovery of the target spatial signal \( u \) from \( v \) can be guided by another available spatial signal. An example is provided by the guided depth map super-resolution problem, where the high resolution depth map \( u \) must be recovered from the low resolution depth map \( v \) when a high resolution image \( z \), aligned with \( u \) and with its same resolution, is available. In general it is not very likely to observe sudden depth changes, hence strong depth
map edges in \( u \), within the same object. Moreover, most of the objects are characterized by a main color. As a consequence, a strong edge in the depth map \( u \) is quite unlikely to occur when the same area in the guide image \( z \) is smooth, while it may appear in the presence of an edge in \( z \). Encoding this prior information in the regularizer \( g(\cdot) \) can prevent bleeding effects in the depth map \( u \). This is the case of \textit{Nagel-Enkelmann (NE) like} regularizers [4, 5], for which we provide the formulation of [5] as an example:

\[
NE(u) = \sum_i \left\| \begin{pmatrix} \exp(-\gamma \| \nabla z(i) \|_2^\beta) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \nabla z(i)^\top \\ \nabla z(i)^\top/\| \nabla z(i) \|_2 \end{pmatrix} \right\|_2^2 \|
\]

(1.5)

where \( \nabla^\perp z(i) \in \mathbb{R}^2 \) is one of the two vectors orthogonal to \( \nabla z(i) \in \mathbb{R}^2 \), \( \gamma \) and \( \beta > 0 \) are tunable parameters and we recall that \( z \) represents the guide image while \( u \) the target high resolution depth map, both at the same resolution. In the presence of a strong guide image edge at the pixel \( i \), i.e., ideally for \( \| \nabla z(i) \|_2 \to +\infty \), NE tends to zero for both \( \nabla u(i) \) aligned with \( \nabla z(i) \) and \( \nabla u(i) \to 0 \). Equivalently, in the presence of an edge in the guide image, an edge in the depth map is considered possible; in particular, the edge in \( u \) is not promoted, but, if present, it must be aligned with the guide image edge in order not to incur a penalization. On the other hand, in the absence of a guide image edge at the pixel \( i \), i.e., for \( \| \nabla z(i) \|_2 \to 0 \), the regularizer NE penalizes an edge in the depth map at the pixel \( i \) more and more as its strength increases. Equivalently, in the absence of an edge in the guide image, an edge in the depth map is penalized.

### 1.2 Graph-based regularization

TV and TGV regularizers on one side, and NE-like regularizers on the other one, represent both effective tools. However, mixing these two classes into a single regularizer, in order to take advantage of the properties of both, is either not straightforward or results in not easily interpretable regularizers [6]. The simple spatial gradient \( \nabla(\cdot) \) underneath the presented regularizers limits their applicability. In fact, a signal may not be piece-wise constant when living in its native 2D Euclidean grid, but it may fit the piece-wise constant assumption better when placed on a different topology. This is the main intuition behind graph-based regularization. Graph-based regularization relates to \textit{Graph Signal Processing (GSP)} [7], whose aim is to generalize traditional signal processing on Euclidian domains to arbitrary graph domains.

In graph-based regularization, and more generally in GSP each pixel \( i \) of \( u \in \mathbb{R}^N \) is modeled as a vertex of a weighted graph \( G = (V, E, w) \), where \( V \) is the set of the \( N \) graph vertexes, or equivalently pixels, \( E \) is the set of the graph edges and \( w \) is a function assigning a positive weight to each edge, defined as follows:

\[
w : E \subseteq (V \times V) \to \mathbb{R}, \quad (i, j) \mapsto w(i, j) \in \mathbb{R}_{>0}.
\]
Chapter 1. Introduction

Edges in the graph are meant to connect those pixels \( i \) and \( j \) whose values \( u(i) \) and \( u(j) \), e.g., intensity or depth, are considered as highly correlated. The level of correlation between the pixels \( i \) and \( j \) is encoded in the weight \( w(i,j) \) of the edge \((i,j)\) from \( i \) to \( j \). Moreover, since the graph can potentially be directed, \( u(i,j) \) does not need to match \( w(j,i) \), which permits an even more general interpretation than traditional correlation. Moreover, the graph permits to go beyond the simple spatial correlation assumed in the traditional regularizers presented previously, as it permits to connect pixels which would have been far away from each other on their native 2D pixel grid, albeit potentially highly correlated. In fact, graph-based regularization shares the flavour of Non Local Means (NLM) denoising [8].

In the GSP perspective it is possible to generalize the concept of gradient to signals defined on graphs [7]. In particular, the gradient of the signal \( u \), now modeled as the graph \( \mathcal{G} \), is generalized as follows:

\[
\nabla_{\mathcal{G}} u(i) = \begin{pmatrix}
\sqrt{w(i,j_1)}(u(j_1) - u(i)) \\
\sqrt{w(i,j_2)}(u(j_2) - u(i)) \\
\vdots \\
\sqrt{w(i,j_{K_i})}(u(j_{K_i}) - u(i))
\end{pmatrix},
\]

where \( \{j_1, j_2, \ldots, j_{K_i}\} = \{j \sim i\} \), with \( K_i \in \mathbb{N} \), is the set of pixels such that \((i,j) \in \mathcal{E}\) and the quantity \(\sqrt{w(i,j)}(u(j) - u(i))\) is defined as the derivative of \( u \) along the edge \((i,j)\). The gradient \(\nabla_{\mathcal{G}}\) permits to generalize traditional regularizer such as TV and TGV to the graph setting [9] [10]. As an example, replacing \(\nabla(\cdot)\) with \(\nabla_{\mathcal{G}}(\cdot)\) in Eq. (1.3) leads to the Non Local Total Variation (NLTV) regularizer [9]:

\[
NLTV(u) = \sum_i \|\nabla_{\mathcal{G}} u(i)\|_2. \tag{1.6}
\]

On the one hand, the NLTV regularizer in Eq. (1.6) reduces to the TV regularizer in Eq. (1.3) for a specific choice of the graph and its weights, i.e., building the weights on the sole Euclidean distance between pixels. On the other hand, since the weights of the graph underneath \( u \) can be defined based on a guide image \( z \), graph-based regularizer permit to easily mix regularizers such as TV and TGV with edge aware regularizers such as the NE-like ones.

We observe that graph-based regularization is similar in flavor to Markov Random Fields (MRFs), widely used in the imagining literature. However, while MRFs require the value at each pixel of the target signal \( u \) to be discrete, which clearly limits the target signal accuracy in tasks such as depth estimation, the proposed graph-based framework leads directly to continuous optimization. In fact, in this thesis we investigate the use of graph-based regularization within some relevant inverse problems where the target signal is inherently continuous. We leverage the flexibility of the graph-based regularization to capture the geometry of the target signal in each considered problems and show that graph-based regularizers can lead to state-of-the-art results. We start by extending a simple, yet effective, smooth graph-based regularizer to a
first application in imaging. Then we progressively transition to nonsmooth graph-based regularizers for the latter applications, as nonsmooth regularizers preserve sharp edges and fine details better than their smooth counterpart.

1.3 Thesis outline

In Chapter 2 we address the light field spatial super-resolution problem. Light field cameras can capture the 3D information in a scene with a single exposure. This special feature makes light field cameras very appealing for a variety of applications, from post-capture refocus to depth estimation and image-based rendering. However, light field cameras suffer by design from strong limitations in their spatial resolution. On the one hand, off-the-shelf super-resolution algorithms are not ideal for light field data, as they do not consider its structure. On the other hand, the few super-resolution algorithms explicitly tailored for light field data exhibit significant limitations, such as the need to carry out a costly disparity estimation procedure with sub-pixel precision, or the need for too strong assumptions on the light field geometry, or simply the need of a large amount of light field data in order to train a model. Our contribution is a new light field super-resolution algorithm meant to address these limitations. We use the complementary information in the different light field views to augment the spatial resolution of the whole light field at once. In particular, we show that coupling the multi-view approach with a smooth graph-based regularizer, which enforces the light field geometric structure, permits to avoid the need of a precise and costly disparity estimation step. We provide extensive experiments and show that the proposed framework compares favorably to state-of-the-art light field super-resolution algorithms, both visually and in terms of reconstruction error.

Chapter 3 provides a transition from smooth graph-based regularization to the nonsmooth one. In particular, we extend a specific nonsmooth graph-based regularizer to two applications in imaging. The first application revolves again around light field super-resolution and we show that the proposed nonsmooth regularizer can lead to higher quality results, in terms of sharper edges and finer details, than the regularizer proposed in Chapter 2. The second application is disparity estimation in omnidirectional stereo systems, where the two captured images and the target disparity map live on a spherical surface, hence a graph-based regularizer can be used to model the non trivial correlation underneath each signal.

In Chapter 4 we propose a new algorithm for the joint refinement of a depth map and the estimation of the corresponding normal map. Both the depth maps from active depth sensing devices, such as ToF cameras, and those from stereo and Multi-View Stereo (MVS) methods rely on a depth refinement step as a post processing. In fact, ToF cameras are sensitive to solar interference as well as multi path interference, which can cause noisy depth maps. Similarly, stereo and MVS methods can fall short in occluded or untextured areas, where the matching becomes impossible or ambiguous, respectively. Based on the observation that most human made environments are piece-wise planar or can be well approximated as such, we present
Chapter 1. Introduction

a new graph-based regularizer which explicitly models the scene as piece-wise planar. In particular, we model the target refined depth map as a graph, where strongly connected pixels are assumed to correspond to point of the scene belonging to the same planar surface. The proposed regularizer fits a plane at each pixel based on the depth at its neighboring pixel and enforces the planes estimated at neighboring pixels to be consistent with each other. The explicit fitting of a plane at each pixel leads to the estimation of the scene normals, which can be useful both in the context of scene understanding or in the context of 3D reconstruction, where normals can be used to filter outliers when fusing multiple point clouds. We provide extensive experiments and show the benefits of our method over state-of-the-art depth refinement methods.

Finally, Chapter 5 wraps up the thesis; it provides some perspectives on the open problems in each one of the addressed applications and also some possible research directions.
2 Geometrically consistent light field spatial super-resolution

We live in a 3D world but the pictures taken with traditional cameras can capture just 2D projections of this reality. The light field is a model that has been originally introduced in the context of image-based rendering with the purpose of capturing richer information in a 3D scene [11]. The light emitted by the scene is modeled in terms of rays, each one characterized by a direction and a radiance value. The light field function provides, at each point in space, the radiance from a given direction. The rich information captured by the light field function could be used in many applications, from post-capture refocus to depth estimation or virtual reality.

However, the light field is a theoretical model: in practice the light field function has to be properly sampled, which is a challenging task. A straightforward but hardware-intensive approach relies on camera arrays [12]. In this setup, each camera records an image of the same scene from a particular position and the light field takes the form of an array of views. More recently, the development of the first commercial light field cameras [13, 14] has made light field sampling more accessible. In light field cameras, a micro lens array placed between the main lens and the sensor permits to virtually partition the main lens into sub-apertures, whose images are recorded altogether in a single exposure [15, 16]. As a consequence, a light field camera behaves as a compact camera array, providing multiple images of a 3D scene from slightly different points of view. An example of light field camera architecture is depicted in Figure 2.1.

Even if light field cameras become very appealing, they still face the so called spatio-angular resolution tradeoff. Since the whole array of views is captured by a single sensor, a dense sampling of the light field in the angular domain (i.e., a large number of views) necessarily translates into a sparse sampling in the spatial domain (i.e., low resolution views) and vice versa. A dense angular sampling is at the basis of any light field application, as the 3D information provided by the light field data comes from the availability of different views. It follows that the angular sampling cannot be excessively penalized to favor the spatial resolution. Moreover, even in the limit scenario of a light field with just two views, the spatial resolution of each one may be reduced to half of the sensor [15]. Consequently, the light field
Chapter 2. Geometrically consistent light field spatial super-resolution

![Diagram of light field camera architecture](image)

**Figure 2.1** – Architecture of the light field camera 1.0 (figure courtesy of [17]). The figure on the left-hand side sketches the camera architecture. A micro lens array (MLA) is placed between the main lens (ML) and the sensor (Chip). Each micro lens plays the role of a traditional camera pixel. However, rather than just integrating the received light rays, the micro lens directs rays coming from different areas (equivalently, sub-apertures) of the main lens to the different pixels in the portion of the sensor underneath. As an example, the portion of the sensor $m_2$ records the different rays reaching the corresponding micro lens: rays depicted with different colors reach the micro lens from different sub-apertures of the main lens, hence they are recorded at different pixels. The figure on the right-hand side outlines the procedure that extracts the light field views from the sensor. The portions of the sensor $m_1$, $m_2$ and $m_3$ correspond to different micro lenses: all the pixels represented with the same color are associated to the same main lens sub-aperture, hence they form a so-called view.

views exhibit a significantly lower resolution than images from traditional cameras, and many light field applications, such as depth estimation, happen to be very challenging on low spatial resolution data. The design of spatial super-resolution techniques, aiming at increasing the view resolution, is therefore crucial in order to fully exploit the potential of light field cameras.

In this chapter\(^1\) we propose a new light field super-resolution algorithm that augments the resolution of all the views together, while relying only on an approximate disparity estimation procedure. In particular, we propose to cast light field spatial super-resolution into a global optimization problem, whose objective function is designed to capture the relations between the light field views. The objective function comprises three terms. The first one enforces data fidelity, by constraining each high resolution view to be consistent with its low resolution counterpart. The second one is a warping term, which gathers for each view the complementary information encoded in the other ones. The third one is a graph-based regularizer, which regularizes the high resolution views by enforcing smoothness along the light field epipolar lines that define the light field geometric structure. These terms altogether form a quadratic objective function that we minimize iteratively with the Proximal Point Algorithm. The results show that our algorithm compares favorably to state-of-the-art light field super-resolution algorithms, both visually and in terms of reconstruction error.

The chapter is organized as follows. Section 2.1 presents an overview of the related literature.

---

\(^1\)The work presented in this chapter has been published in [18, 19].
Section 2.2 formalizes the light field structure. Section 2.3 presents our problem formulation and carefully analyzes each of its terms. Section 2.4 provides a detailed description of our super-resolution algorithm. Section 2.5 is dedicated to our experiments. Finally, Section 2.6 concludes the chapter.

2.1 Related work

The super-resolution literature is quite vast, but it can be divided mainly into two areas: single-frame and multi-frame super-resolution methods. In single-frame super-resolution, only one image from a scene is provided and its resolution has to be increased. This goal is typically achieved by learning a mapping from the low resolution data to the high resolution one, either on an external training set [20–22] or on the image itself [23, 24]. Single-frame algorithms can be applied to each light field view separately in order to augment the resolution of the whole light field, but this approach would neither exploit the high correlation among the views, nor enforce the consistency among them.

In the multi-frame scenario, multiple images of the same scene are used to increase the resolution of a target image. To this purpose, a global image warping model is assumed, where all the available images are treated as translated and rotated versions of the target one [25, 26]. The multi-frame super-resolution scenario resembles the light field one, but its global image warping model does not fit the light field structure. In particular, the different moving speeds of the objects in the scene across the light field views, which encode their different depths, cannot be captured by a global warping model. Multi-frame algorithms employing more complex warping models exist, for example in video super-resolution [27, 28], yet the warping models do not exactly fit the geometry of the light field data and their construction is computationally demanding. In particular, multi-frame video super-resolution involves two main steps, namely optical flow estimation, which finds correspondences between temporally successive frames, and eventually a super-resolution step that is built on the optical flow.

In the light field representation, the views lie on a two-dimensional grid with adjacent views sharing a constant baseline under the assumption of both vertical and horizontal registration. As a consequence, not only the optical flow computation reduces to disparity estimation, but also the disparity map at one view determines its warping to every other view in the light field, in the absence of occlusions. In [29] Wanner and Goldluecke build over these observations to extract the disparity map at each view directly from the epipolar line slopes with the help of a structure tensor operator. The estimated low resolution disparity maps are upscaled and, similarly to multi-frame super-resolution, all the views are projected to the target one within a global optimization formulation endowed with a Total Variation (TV) regularizer. Although the structure tensor operator permits to carry out disparity estimation with continuous precision, this task remains very challenging at low spatial resolution. As a result, the disparity errors translate into significant artifacts in the textured areas and along the object edges of the super-resolved target view. Finally, each light field view is super-resolved separately in [29],
Chapter 2. Geometrically consistent light field spatial super-resolution

which does not permit to fully exploit the inter-view dependencies.

In another work, Heber and Pock [30] consider the matrix obtained by warping all the views to a reference one, and they propose to model it as the sum of a low rank matrix and a noise one, where the later describes the noise and occlusions. This model, that resembles Robust PCA [31], is primarily meant for disparity estimation at the reference view. However, the authors show that a slight modification of the objective function can provide the corresponding high resolution view, in addition to the low resolution disparity map at the reference view. The algorithm could ideally be applied separately to each view in order to super-resolve the whole light field, but again this would not permit to fully exploit the inter-view dependencies.

In a different framework, Mitra and Veeraraghavan propose a light field super-resolution algorithm based on a learning procedure [32]. Each view in the low resolution light field is divided into patches that are possibly overlapping. All the patches at the same spatial coordinates in the different views form a light field with very small spatial resolution, i.e., a light field patch. The authors assign a constant disparity to each light field patch, i.e., all the objects within the light field patch are assumed to lie at the same depth in the scene. A different Gaussian Mixture Model (GMM) prior for high resolution light field patches is learnt offline for each discrete disparity value, and it is then employed within a MAP estimator to super-resolve each light field patch with the corresponding disparity. However, the learning strategy has some drawbacks: the dependency of the reconstruction on the chosen training set, the need for a new training for each super-resolution factor, and finally the need for a proper discretization of the disparity range, which introduces a tradeoff between the reconstruction quality and the time required by both the training and the reconstruction steps. Moreover, the simple assumption of constant disparity within each light field patch leads to severe artifacts at depth discontinuities in the super-resolved light field views.

The light field super-resolution problem has been addressed within the framework of Convolutional Neural Networks (CNNs) too. In particular, Yoon et al. [33] consider the cascade of two CNNs, the first meant to super-resolve the given light field views, and the second to synthesize new high resolution views based on the previously super-resolved ones. However, the first CNN (whose design is borrowed from [22]) is meant for single-frame super-resolution, therefore the views are super-resolved independently, without considering the light field structure.

Finally, we note that some authors, e.g., Bishop et al. [34], consider the recovery of an all-in-focus image with full sensor resolution from the light field camera output. They refer to this task as light field super-resolution although it is different from the problem considered in this chapter. In fact, in this chapter no light field applications is considered a priori: the light field views are all super-resolved, thus enabling any light field application to be performed later at a resolution higher than the original one. Differently from the other light field super-resolution algorithms, our new method does not rely on a precise and costly disparity estimation step, and it does not rely on a learning procedure. Moreover, our algorithm reconstructs all the
views jointly, provides homogeneous quality across the reconstructed views, and it preserves the light field structure.

2.2 Light field structure

In the light field literature, it is common to parametrize the light rays from a 3D scene by the coordinates of their intersection with two parallel planes, typically referred to as the spatial plane \( \Omega \) and the angular plane \( \Pi \). Each light ray is associated to a radiance value, and a pinhole camera with its aperture on the plane \( \Pi \) and its sensor on the plane \( \Omega \) can record the radiance of all those rays accommodated by its aperture. This is represented in Figure 2.2, where each pinhole camera is represented as a pyramid, with its vertex and basis representing the camera aperture and sensor, respectively. In general, an array of pinhole cameras can perform a regular sampling of the angular plane \( \Pi \), therefore the sampled light field takes the form of a set of images captured from different points of view. This is the sampling scheme approximated by both camera arrays and light field cameras.

In the following we consider the light field as the output of an \( M \times M \) array of pinhole cameras, each one equipped with an \( N \times N \) pixel sensor. Each camera is identified through the angular coordinates \((s, t)\) with \( s, t \in \{1, 2, \ldots, M\} \), while a pixel within the camera sensor is identified through the spatial coordinates \((x, y)\) with \( x, y \in \{1, 2, \ldots, N\} \). The distance between the apertures of horizontally or vertically adjacent cameras is \( b \), referred to as the baseline. The distance between the planes \( \Pi \) and \( \Omega \) is \( f \), referred to as the camera focal length. Figure 2.2 sketches two cameras of the \( M \times M \) array. Within this setup, we can represent the light field as an \( N \times N \times M \times M \) real tensor \( U \), with \( U(x, y, s, t) \) the intensity of the pixel with coordinates \((x, y)\) in the view of the camera at \((s, t)\). In particular, we denote the view at \((s, t)\) as \( U_{s, t} = U(\cdot, \cdot, s, t) \in \mathbb{R}^{N \times N} \). Finally, without loss of generality, we assume that each pair of horizontally or vertically adjacent views in the light field are registered.

With reference to Figure 2.2, we now describe in more details the particular structure of the light field data. We consider a point \( P \in \mathbb{R}^3 \) at depth \( z \) from \( \Pi \), whose projection on one of the cameras is represented by the pixel \( U_{s, t}(x, y) \), in the right view of Figure 2.2. We now look at the projection of \( P \) on the other views \( U_{s, t'} \) in the same row of the camera array, such as the left view in Figure 2.2. We observe that, in the absence of occlusions and under the Lambertian assumption (i.e., all the rays emitted by the point \( P \) exhibit the same radiance), the projection of \( P \) obeys the following stereo equation:

\[
U_{s, t}(x, y) = U_{s, t'}(x, y + (t - t')d_{x, y}) = U_{s, t'}(x, y') \tag{2.1}
\]

where \( d_{x, y} = D_{s, t}(x, y) = f b / z \), with \( D_{s, t} \in \mathbb{R}^{N \times N} \) the disparity map of the view \( U_{s, t} \) with respect to its left view \( U_{s, t - 1} \). A visual interpretation of Eq. (2.1) is provided by the Epipolar Plane Image (EPI) [35] in Figure 2.3b, which represents a slice \( U(x, \cdot, s, \cdot)^\top \in \mathbb{R}^{M \times N} \) of the light field.
Chapter 2. Geometrically consistent light field spatial super-resolution

Figure 2.2 – Light field sampling in the two plane parametrization. The light field is ideally sampled through an $M \times M$ array of pinhole cameras. The pinhole cameras at coordinates $(s, t)$ and $(s, t')$ in the camera array are represented as two pyramids, with their apertures denoted by the two green dots on the plane $\Pi$, and their $N \times N$ sensors represented by the two orange squares on the plane $\Omega$. The distance between the two planes is the focal length $f$. The distance between the apertures of horizontally or vertically adjacent views in the $M \times M$ array is the baseline $b$, therefore the distance between the two green dots on the plane $\Pi$ is $|(t - t')b|$. The small yellow squares in the two sensors denote the pixel $(x, y)$. The pixel $(x, y)$ of the camera at $(s, t)$ captures one of the light rays (in red) emitted by a point $P$ at depth $z$ in the scene. The disparity associated to the pixel $(x, y)$ of the camera at $(s, t)$ is $d_{x,y}$, therefore the projection of $P$ on the sensor of the camera at $(s, t')$ lies at $(x, y') = (x, y + (t - t')d_{x,y})$. The intersection point $(x, y')$ is denoted by a red spot, as it does not necessarily correspond to a pixel, since $d_{x,y}$ is not necessarily integer.

This exhibits a clear line pattern, as the projection $U_{s,t}(x, y)$ of the point $P$ moves at a constant speed across the other views $U_{s,t'}$, with its speed determined by $d_{x,y}$. We stress out that, although $U_{s,t}(x, y)$ is a pixel in the captured light field, all its projections $U_{s,t'}(x, y')$ do not necessarily correspond to actual pixels in the light field views, as $y'$ may not be integer. We finally observe that Eq. (2.1) can be extended to the whole light field:

$$U_{s,t}(x, y) = U_{s',t'}(x + (s - s')d_{x,y}, y + (t - t')d_{x,y})$$
$$= U_{s',t'}(x', y').$$

(2.2)
2.3 Problem formulation

We refer to the model in Eq. (2.2) as the light field structure.

Later on, for the sake of clarity, we will denote a light field view either by its angular coordinates \((s, t)\) or by its linear coordinate \(k = ((t - 1)M + s) \in \{1, 2, \ldots, M^2\}\). In particular, we have \(U_{s,t} = U_k\) where \(U_k\) is the \(k\)-th view encountered when visiting the camera array in column major order. We also handle the light field in a vectorized form, with the following notation:

- \(u_{s,t} = u_k \in \mathbb{R}^{N^2}\) is the vectorized form of the view \(U_{s,t}\),
- \(u = [u_1^T, u_2^T, \ldots, u_{M^2}^T]^T \in \mathbb{R}^{N^2M^2}\),

where the vectorized form of a matrix is simply obtained by visiting its entries in column major order.

2.3 Problem formulation

The light field (spatial) super-resolution problem concerns the recovery of the high resolution light field \(U\) from its low resolution counterpart \(V\) at resolution \((N/\alpha) \times (N/\alpha) \times M \times M\), with \(\alpha \in \mathbb{N}\) the super-resolution factor. Equivalently, we aim at super-resolving each view \(V_{s,t} \in \mathbb{R}^{(N/\alpha)^2}\) to get its high resolution version \(U_{s,t} \in \mathbb{R}^{N^2}\). In order to recover the high resolution light field from the low resolution data, we propose to minimize the following objective function:

\[
\begin{aligned}
\mathbf{u}^* \in \arg\min_u \quad & f_1(u) + \lambda_2 f_2(u) + \lambda_3 g(u) \\
\text{subject to} \quad & h(u) = 0
\end{aligned}
\]  

(2.3)

where each term describes one of the constraints about the light field structure and the multipliers \(\lambda_2\) and \(\lambda_3\) balance the different terms. We now analyze each one of them separately.

Each pair of high and low resolution views have to be consistent and we model their relationship as follows:

\[
v_k = SBu_k + n_k
\]  

(2.4)

where \(B \in \mathbb{R}^{N^2 \times N^2}\) and \(S \in \mathbb{R}^{(N/\alpha)^2 \times N^2}\) denote a blurring and a sampling matrix, respectively, and the vector \(n_k \in \mathbb{R}^{(N/\alpha)^2}\) captures possible inaccuracies of the assumed model. The first term in Eq. (2.3) enforces the constraint in Eq. (2.4) for each high resolution and low resolution view pair, and it is typically referred to as the data fidelity term:

\[
f_1(u) = \sum_k \|SBu_k - v_k\|_2^2.
\]  

(2.5)

Then, the various low resolution views in the light field capture the scene from slightly different perspectives, therefore details dropped by digital sensor sampling at one view may survive in another one. Gathering at one view all the complementary information from the others can augment its resolution. This can be achieved by enforcing that the high resolution view \(u_k\)
Figure 2.3 – Example of light field and Epipolar Plane Image (EPI). Figure (a) shows an array of $3 \times 3$ views, extracted from the knights light field (Stanford dataset [36]) which actually consists of an array of $17 \times 17$ views. Figure (b) shows an EPI from the original $17 \times 17$ knights light field. In particular, the $t$-th row in the EPI corresponds to the row $U_{t,\cdot}(730, \cdot)$. The top, central, and bottom red dashed rows in (b) correspond to the left, central, and right dashed rows in red in the sample views in (a), respectively.

can generate all the other low resolution views $v_{k'}$ in the light field, with $k' \neq k$. For every view $u_k$ we thus have the following model:

$$v_{k'} = SBF_k^{k'} u_k + n_{k'}^{k'}, \quad \forall k' \neq k$$

(2.6)

where the matrix $F_k^{k'} \in \mathbb{R}^{N_x \times N_y}$ is such that $F_k^{k'} u_k = u_{k'}$ and it is typically referred to as a warping matrix. The vector $n_{k'}^{k'}$ captures possible inaccuracies of the model, such as the presence of pixels of $v_{k'}$ that cannot be generated because they correspond to occluded areas in $u_k$. The second term in Eq. (2.3) enforces the constraint in Eq. (2.6) for every high resolution view:

$$f_2(u) = \sum_{k \in \mathcal{K}} \sum_{k' \in \mathcal{K} \setminus \{k\}} \| H_k^{k'} \left( SBF_k^{k'} u_k - v_{k'} \right) \|_2^2$$

(2.7)
2.3. Problem formulation

where the matrix $H_k^l \in \mathbb{R}^{(N/\alpha)^2 \times (N/\alpha)^2}$ is diagonal and binary, and it masks those pixels of $v_k^l$ that cannot be generated due to occlusions in $u_k$, while $\mathcal{N}_k^+$ denotes a subset of the views (potentially all) with $k \notin \mathcal{N}_k^+$.

Finally, a regularizer $g(\cdot)$ happens to be necessary in the overall objective function of Eq. (2.3), as the original problem in Eq. (2.4), and encoded in term $f_1(\cdot)$, is ill-posed due to the fat matrix $S$. The second term $f_2(\cdot)$ can help, but the warping matrices $F_k^l$ in Eq. (2.7) are not known exactly, such that the third term $g(\cdot)$ is necessary. We adopt a graph-based regularizer and define $g(\cdot)$ as follows:

$$g(u) = u^\top L u \quad (2.8)$$

where the positive semi-definite matrix $L \in \mathbb{R}^{(M N)^2 \times (M N)^2}$ is the un-normalized Laplacian [7] of an undirected weighted graph designed to capture the light field structure. In particular, each pixel in the high resolution light field is modeled as a vertex in a graph, where the edges connect each pixel to its projections on the other views. The quadratic form in Eq. (2.8) enforces connected pixels to share similar intensity values, thus promoting the light field structure described in Eq. (2.2).

The graph can be represented through its adjacency matrix $W \in \mathbb{R}^{N^2 M^2 \times N^2 M^2}$, where each row corresponds to a pixel:

$$W(i, j) = \begin{cases} w(i, j) & \text{if } (i, j) \in \mathcal{E} \\ 0 & \text{otherwise,} \end{cases}$$

where we recall that $\mathcal{E}$ is the set of edges in the graph $\mathcal{G}$. Since the graph is assumed to be undirected, the adjacency matrix is symmetric: $W(i, j) = W(j, i)$. Finally, we can rewrite the term $g(\cdot)$ in Eq. (2.8) as follows:

$$g(u) = \frac{1}{2} \sum_{j \sim i} W(i, j) (u(j) - u(i))^2 \quad (2.9)$$

where $j \sim i$ denotes the set of the vertices $j$ directly connected to the vertex $i$ and we recall that the scalar $u(i)$ is the $i$–th entry of the vectorized light field $u$. In Eq. (2.9) the term $g(\cdot)$ penalizes significant intensity variations along highly weighted edges. A weight typically captures the similarity between two vertices, therefore the minimization of Eq. (2.9) leads to an adaptive smoothing [7], ideally along the EPI lines of Figure 2.3b in our light field framework.

Differently from the other light field super-resolution methods, the proposed formulation permits to address the recovery of the whole light field altogether, thanks to the global regularizer $g(\cdot)$. The term $f_2(\cdot)$ permits to augment the resolution of each view without recurring to external data and learning procedures. However, differently from video super-resolution or the light field super-resolution approach in [29], the warping matrices in $f_2(\cdot)$ rely only on a rough estimation of the disparity at each view. This is possible thanks to the graph regularizer $g(\cdot)$, which acts on each view as a denoising term based on nonlocal similarities [37] but at the same time enforces the full light field structure captured by the graph.
2.4 Super-resolution algorithm

We now describe the algorithm that we use to solve the optimization problem in Eq. (2.3). We first discuss the construction of the warping matrices of the term $f_2(\cdot)$ in Eq. (2.7) and then the construction of the graph in the regularizer $g(\cdot)$ in Eq. (2.8). Finally, we describe the complete super-resolution algorithm.

2.4.1 Warping matrix construction

We define the set of the neighboring views $\mathcal{N}_k^+$ in the term $f_2(\cdot)$ in Eq. (2.7) as containing only the four views $U_{k'}$ adjacent to $U_k$ in the light field:

$$\{U_{k'} : k' \in \mathcal{N}_k^+\} = \{U_{s,t\pm1}, U_{s\pm1,t}\}.$$
We introduce the following function, which assigns a score to each integer disparity value

\[ d \]

with the highest score:

\[ \argmax_{d \in \mathcal{N}_k^+} \xi(d) \]  

This choice reduces the number of the warping matrices but at the same time does not limit our problem formulation, as the interlaced structure of the term \( f_2(\cdot) \) constrains together also those pairs of views that are not explicitly constrained in \( f_2(\cdot) \).

The inner summation in Eq. (2.7) considers the set of the four warping matrices \( \{ F^k_k : k' \in \mathcal{N}_k^+ \} \) that warp the view \( U_k \) to each one of the four views \( U_k' \). Conversely, but without loss of generality, in this section we consider the set of the four warping matrices \( \{ F^k_k : k' \in \mathcal{N}_k^+ \} \) that warp each one of the four views \( U_k' \) to the view \( U_k \). The warping matrix \( F^k_k \) is such that \( F^k_k \) computes the pixel \( u_k(i) = U_k(x, y) = U_{k',t}(x, y) \) as a convex combination of those pixels around its projection on \( U_k = U_{k',t} \). Note that the convex combination is necessary, as the projection does not lie at integer spatial coordinates in general. The exact position of the projection on \( U_{k',t} \) is determined by the disparity value \( d_{x,y} \) associated to the pixel \( U_{k',t}(x, y) \). This is represented in Figure 2.4, which shows that the projections of \( U_{k',t}(x, y) \) on the four neighboring views lie on the edges of a virtual square (in red) centered on the pixel \( U_{k',t}(x, y) \) and whose size depends on the disparity value \( d_{x,y} \).

We estimate roughly the disparity value \( d_{x,y} \) by finding a \( \delta \in \mathbb{Z} \) such that \( d_{x,y} \in [\delta, \delta + 1] \). In details, we first define a similarity score between the target pixel \( U_{s,t}(x, y) \) and a generic pixel \( U_{s',t'}(x', y') \) as follows:

\[
\rho_{s',t'}(x', y') = \exp \left( -\frac{\|Q_{s,t}(x, y) - Q_{s',t'}(x', y')\|^2_F}{\sigma^2} \right) \tag{2.10}
\]

where \( Q_{s,t}(x, y) \in \mathbb{R}^{Q \times Q} \) is a square patch centered at the pixel \( U_{s,t}(x, y) \), \( \cdot\cdot\cdot \) denotes the Frobenius norm and \( \sigma \) is a tunable constant. Then, we center a search window at \( U_{s',t'}(x, y) \) in each one of the four views with \( k' \in \mathcal{N}_k^+ \), as represented in Figure 2.4. In particular, we consider:

- a \( 1 \times C \) pixel window for \( (s', t') = (s, t \pm 1) \),
- a \( C \times 1 \) pixel window for \( (s', t') = (s \pm 1, t) \),

with \( C \in \mathbb{N} \) the disparity range assumed for the whole light field, i.e., \( d_{x,y} \in [-C/2, C/2] \).

We introduce the following function, which assigns a score to each integer disparity value \( d \in \{-C/2, \ldots, C/2\} \):

\[
\xi(d) = \rho_{s,t-1}(x, y + d) + \rho_{s,t+1}(x, y - d) + \rho_{s-1,t}(x + d, y) + \rho_{s+1,t}(x - d, y).
\]

In order to estimate \( \delta \in \mathbb{Z} \) such that \( d_{x,y} \in [\delta, \delta + 1] \), we first compute the disparity value \( d^*_1 \) with the highest score:

\[
d^*_1 = \argmax_{d \in [-C/2, \ldots, C/2]} \xi(d). \tag{2.11}
\]
We select \(d_1^*\) to be one of the two extrema of \([\delta, \delta+1]\). We select the other extremum, denoted with \(d_2^*\), as the disparity value with the highest score between \(d_1^* - 1\) and \(d_1^* + 1\):

\[
d_2^* = \arg\max_{d \in [d_1^*-1, d_1^*+1]} \xi(d),
\]

where we assume \(\xi(d) = 0\) for \(d \notin [-C/2, \ldots, C/2]\). Finally, since \(d_1^*\) and \(d_2^*\) are the two extrema of \([\delta, \delta+1]\), we define \(\delta\) as follows:

\[
\delta = \min\{d_1^*, d_2^*\}.
\]

We can now fill the \(i\)-th row of the matrix \(F_k\) such that the pixel \(u_k(i) = U_{s,t}(x, y)\) is computed as the convex combination of the two closest pixels to its projection on \(U_k = U_{s', t'}\), namely the following two pixels:

\[
\{U_{k'}(x, y + \delta), U_{k'}(x, y + \delta + 1)\} \text{ for } (s', t') = (s, t-1),
\]

\[
\{U_{k'}(x, y - \delta - 1), U_{k'}(x, y - \delta)\} \text{ for } (s', t') = (s, t+1),
\]

\[
\{U_{k'}(x+\delta, y), U_{k'}(x+\delta + 1, y)\} \text{ for } (s', t') = (s-1, t),
\]

\[
\{U_{k'}(x-\delta - 1, y), U_{k'}(x-\delta, y)\} \text{ for } (s', t') = (s+1, t),
\]

which are indicated in green in Figure 2.4. Once the two pixels involved in the convex combination at the \(i\)-th row of the matrix \(F_k\) are determined, the \(i\)-th row can be constructed. As an example, let us focus on the left neighboring view \(U_{k'} = U_{s,t-1}\). The two pixels involved in the convex combination at the \(i\)-th row of the matrix \(F_k\) are the following:

\[
\{u_{k'}(j_1) = U_{k'}(x, y + \delta), u_{k'}(j_2) = U_{k'}(x, y + \delta + 1)\}.
\]

We thus define the \(i\)-th row of the matrix \(F_k\) as follows:

\[
F_k(i, j) = \begin{cases} 
\rho_{s,t-1}(x, y + \delta) / \rho & \text{if } j = j_1 \\
\rho_{s,t-1}(x, y + \delta + 1) / \rho & \text{if } j = j_2 \\
0 & \text{otherwise}
\end{cases}
\]

with \(\rho = \rho_{s,t-1}(x, y + \delta) + \rho_{s,t-1}(x, y + \delta + 1)\). In particular, each one of the two pixels in the convex combination has a weight that is proportional to its similarity to the target pixel \(U_{s,t}(x, y)\). For the remaining three neighboring views we proceed similarly.

We stress out that, for each pixel \(u_k(i) = U_{s,t}(x, y)\), the outlined procedure fills the \(i\)-th row of each one of the four matrices \(F_k\) with \(k' \in \mathbb{N}^+_k\). As illustrated in Figure 2.4, the pair of pixels selected in each one of the four neighboring views encloses one edge of the red square hosting the projections of the pixel \(U_{s,t}(x, y)\), therefore this procedure contributes to enforce the light field structure in Eq. (2.2). Later on, we will refer to this particular structure as the square constraint.
Finally, since occlusions are mostly handled by the regularizer \( g(\cdot) \), we use the masking matrix \( H_k \) in Eq. (2.7) to handle only the trivial occlusions due to the image borders.

### 2.4.2 Regularization graph construction

The effectiveness of the term \( g(\cdot) \) depends on the graph capability to capture the underlying structure of the light field. Ideally, we would like to connect each pixel \( U_{s,t}(x,y) \) in the light field to its projections on the other views, as these all share the same intensity value under the Lambertian assumption. However, since the projections do not lie at integer spatial coordinates in general, we adopt a procedure similar to the warping matrix construction and we aim at connecting the pixel \( U_{s,t}(x,y) \) to those pixels that are close to its projections on the other views. We thus propose a three step approach to the computation of the graph adjacency matrix \( W \) in Eq. (2.9).

#### Edge weight computation

We consider a view \( U_{s,t} \) and define its set of neighboring views \( \mathcal{N}_k \) as follows:

\[
\mathcal{N}_k = \mathcal{N}_k^+ \cup \mathcal{N}_k^x
\]

where we extend the neighborhood considered in the warping matrix construction with the four diagonal views. In particular, \( \mathcal{N}_k^x \) is defined as follows:

\[
\{ U_{k'} : k' \in \mathcal{N}_k^x \} = \{ U_{s-1,t \pm 1}, U_{s+1,t \pm 1} \}.
\]

The full set of neighboring views is represented in Figure 2.4, with the views in \( \mathcal{N}_k^+ \) in orange, and those in \( \mathcal{N}_k^x \) in green. We then consider a pixel \( u(i) = U_{s,t}(x,y) \) and define its edges toward one neighboring view \( U_{k'} = U_{s',t'} \) with \( k' \in \mathcal{N}_k \). We center a search window at the pixel \( U_{s',t'}(x,y) \) and compute the following similarity score between the pixel \( U_{s,t}(x,y) = u(i) \) and each pixel \( U_{s',t'}(x',y') = u(j) \) in the considered window:

\[
W_A(i,j) = \exp\left( -\frac{\| Q_{s,t}(x,y) - Q_{s',t'}(x',y') \|^2}{\sigma^2} \right), \tag{2.12}
\]

with the notation already introduced in Section 2.4.1. We repeat the procedure for each one of the eight neighboring views in \( \mathcal{N}_k \), but we use differently shaped windows at different views:

- a \( 1 \times C \) pixel window for \((s',t') = (s, t \pm 1)\),
- a \( C \times 1 \) pixel window for \((s',t') = (s \pm 1, t)\),
- a \( C \times C \) pixel window otherwise.

This is illustrated in Figure 2.4. The \( C \times C \) pixel window is introduced for the diagonal views.
Chapter 2. Geometrically consistent light field spatial super-resolution

\[ U_k = U_{s', t'}, \] with \( k' \in \mathcal{N}_k \), as the projection of the pixel \( U_{s, t}(x, y) \) on these views lies neither along the row \( x \), nor along the column \( y \). Iterating the outlined procedure over each pixel \( u(i) \) in the light field leads to the construction of the adjacency matrix \( W_A \). We regard \( W_A \) as the adjacency matrix of a directed graph, with \( W_A(i, j) \) the weight of the edge from \( u(i) \) to \( u(j) \).

**Edge pruning**

We want to keep only the most important connections in the graph. We thus perform a pruning of the edges leaving the pixel \( U_{s, t}(x, y) \) toward the eight neighboring views. In particular, we keep only

- the two largest weight edges, for \((s', t') = (s, t \pm 1)\),
- the two largest weight edges, for \((s', t') = (s \pm 1, t)\),
- the four largest weight edges, otherwise.

For the diagonal neighboring views \( U_{k'} = U_{s', t'}, \) with \( k' \in \mathcal{N}_k \), we allow four weights rather than two as it is more difficult to detect those pixels that lie close to the projection of \( U_{s, t}(x, y) \). We define \( W_B \) as the adjacency matrix after the pruning.

**Symmetric adjacency matrix**

We finally carry out the symmetrization of the matrix \( W_B \) and set \( W = W_B \) in Eq. (2.9). We preserve an edge between two vertexes \( u(i) \) and \( u(j) \) if and only if both the entry \( W_B(i, j) \) and \( W_B(j, i) \) are non zero. If this is the case, then \( W_B(i, j) = W_B(j, i) \) necessarily holds true, and the weights are maintained. This procedure mimics the well-known left-right disparity check of stereo vision [38].

We finally note that, as proposed in [18], the constructed graph can be used to build an alternative warping matrix to the one in Section 2.4.1. We recall that the matrix \( F^k_{k'} \) is such that \( F^k_{k'} u_{k'} \approx u_k \). In particular, the \( i \)-th row of this matrix is expected to compute the pixel \( u_k(i) = U_{s, t}(x, y) \) as a convex combination of those pixels around its projection on \( U_{k'} = U_{s', t'} \). We thus observe that the sub-matrix \( W_S \), obtained by extracting the rows \((k - 1)N^2 + 1, \ldots, kN^2\) and the columns \((k' - 1)N^2, \ldots, k'N^2\) from the adjacency matrix \( W \), represents a directed weighted graph with edges from the pixels of the view \( U_k = U_{s, t} \) (rows of the matrix) to the pixels of the view \( U_{k'} = U_{s', t'} \) (columns of the matrix). In this graph, the pixel \( u_k(i) = U_{s, t}(x, y) \) is connected to a subset of pixels that lie close to its projections on \( U_{k'} = U_{s', t'} \). We thus normalize the rows of \( W_S \) such that they sum up to one, in order to implement a convex combination, and set \( F^k_{k'} = \tilde{W}_S \) with \( \tilde{W}_S \) the normalized sub-matrix. This alternative approach to the warping matrix construction does not take the light field structure explicitly into account, but it represents a valid alternative when computational resources are limited.
2.4. Super-resolution algorithm

2.4.3 Optimization algorithm

We now have all the ingredients to solve the quadratic optimization problem in Eq. (2.3). We observe that it corresponds to a quadratic problem. We rewrite the first term, in Eq. (2.5), as follows:

\[ f_1(u) = \| Au - v \|_2^2 \]

\[ = u^\top A^\top A u - 2 v^\top A u + v^\top v \] (2.13)

with \( A = I \otimes SB, I \in \mathbb{R}^{M^2} \) the identity matrix, and \( \otimes \) the Kronecker product. For the second term, in Eq. (2.7), we introduce the following matrices:

- \( H_k = \text{diag}(H_1^k, H_2^k, \ldots, H_{M^2}^k) \),

- \( F_k = e_k^\top \otimes \begin{bmatrix} (H_1^k A F_k)^\top \\ (H_2^k A F_k)^\top \\ \vdots \\ (H_{M^2}^k A F_k)^\top \end{bmatrix} \),

where diag denotes a block diagonal matrix, and \( e_k \in \mathbb{R}^{M^2} \) denotes the \( k \)-th vector of the canonical basis, with \( e_k(k) = 1 \) and zero elsewhere. The matrices \( H_k^{k'} \) and \( F_k^{k'} \), originally defined only for \( k' \in \mathcal{N}_k^+ \), have been extended to the whole light field by assuming them to be zero for \( k' \notin \mathcal{N}_k^+ \). Finally, it is possible to remove the inner sum in Eq. (2.7):

\[ f_2(u) = \sum_k \| H_k A F_k u - H_k v \|_2^2 \]

\[ = \sum_k u^\top (H_k A F_k)^\top (H_k A F_k) u - 2 (H_k v)^\top (H_k A F_k) u + (H_k A F_k)^\top (H_k v) \] (2.14)

Replacing Eq. (2.13) and Eq. (2.14) in Eq. (2.3) permits to rewrite the objective function \( h(u) \) in a quadratic form:

\[ u^* \in \arg\min_u \frac{1}{2} u^\top P u + q^\top u + r \] (2.15)

with

\[ P = 2 \left( A^\top A + \lambda_2 \sum_k (H_k A F_k)^\top (H_k A F_k) + \lambda_3 I \right), \]

\[ q = -2 \left( A^\top + \lambda_2 \sum_k (H_k A F_k)^\top H_k \right) v, \]

\[ r = v^\top \left( I + \lambda_2 \sum_k H_k^\top H_k \right) v. \]

We observe that, in general, the matrix \( P \) is positive semi-definite, therefore it is not possible
Chapter 2. Geometrically consistent light field spatial super-resolution

Algorithm 1 Graph-Based Light Field Super-Resolution

\textbf{Input: } \(v = [v_1, \ldots, v_{M^2}]\), \(\alpha \in \mathbb{N}, \beta > 0, \text{iter.}\)

\textbf{Output: } \(u = [u_1, \ldots, u_{M^2}]\).

1: \(u \leftarrow \text{bilinear interp. of } v_k \text{ by } \alpha, \forall k = 1, \ldots, M^2;\)
2: \(\text{for } i = 1 \text{ to iter do}\)
3: \(\text{build the graph adjacency matrix } W \text{ on } u;\)
4: \(\text{build the matrices } F_k \text{ on } u, \forall k = 1, \ldots, M^2;\)
5: \(\text{update the matrix } P \text{ and the vector } q;\)
6: \(z \leftarrow u;\) \(\triangleright \text{Initialize CG}\)
7: \(\text{while convergence is reached do}\)
8: \(z \leftarrow \text{CG}(P + (1/\beta), (z/\beta) - q);\)
9: \(\text{end while}\)
10: \(u \leftarrow z;\) \(\triangleright \text{Update } u\)
11: \(\text{end for}\)
12: \(\text{return } u;\)

To solve Eq. (2.15) just by employing the \textit{Conjugate Gradient (CG)} method on the linear system \(\nabla h(u) = Pu - q = 0.\) We thus choose to adopt the \textit{Proximal Point Algorithm (PPA)}, which iteratively solves Eq. (2.15) using the following update rule:

\[
\begin{align*}
\mathbf{u}(i+1) &= \text{prox}_{\beta h(\cdot)} \left( \mathbf{u}(i) \right) \\
&= \arg\min_{\mathbf{u}} h(\mathbf{u}) + \frac{1}{2\beta} \|\mathbf{u} - \mathbf{u}(i)\|^2_2 \\
&= \arg\min_{\mathbf{u}} \frac{1}{2} \mathbf{u}^\top \left( P + \frac{1}{\beta} I \right) \mathbf{u} + \left( \mathbf{q} - \mathbf{u}(i) \right) \mathbf{u}.
\end{align*}
\]

The matrix \(P + (1/\beta) I\) is positive definite for every \(\beta > 0,\) hence we can now use the CG method to solve the linear system \(\nabla \hat{h}(\mathbf{u}) = 0.\) The full \textit{Graph-Based Light Field Super-Resolution} algorithm is summarized in Algorithm 1. We observe that the construction of the warping matrices and the graph requires the high resolution light field to be available. In order to bypass this causality problem, a fast and rough high resolution estimation of the light field is computed via bilinear interpolation at the bootstrap phase. Then, at each new iteration, the warping matrices and the graph can be re-constructed on the new available light field estimate.

The problem in Eq. (2.15) could be solved also with a \textit{Gradient Descent (GD)} algorithm, which is characterized by a less computationally demanding update rule. However, in our experiments, PPA leads to a faster convergence than GD not only in terms of the number of iterations but also in terms of computation time. An analysis of the computational complexity of our algorithm is provided in the Appendix A.1.
2.5 Experiments

2.5.1 Experimental settings

We now provide extensive experiments to analyze the performance of our algorithm. We compare it to the two super-resolution algorithms that, up to our knowledge, are the only ones developed explicitly for light field data, and that we already introduced in Section 2.1: Wanner and Goldluecke [29], and Mitra and Veeraraghavan [32]. We also compare our algorithm to the CNN-based super-resolution algorithm in [22], which represent the state-of-the-art for single-frame super-resolution. Up to the authors knowledge, a multi-frame super-resolution algorithm based on CNNs has not been presented yet.

We test our algorithm on two public datasets: the HCI light field dataset [39] and the (New) Stanford light field dataset [36]. The HCI dataset comprises twelve light fields, each one characterized by a 9 × 9 array of views. Seven light fields have been artificially generated with a 3D creation suite, while the remaining five have been acquired with a traditional SLR camera mounted on a motorized gantry, that permits to move the camera precisely and emulate a camera array with an arbitrary baseline. The HCI dataset is meant to represent the data from a light field camera, where both the baseline distance \( b \) between adjacent views and the disparity range are typically very small. In particular, in the HCI dataset the disparity range is within \([-3, 3]\) pixels. Differently, the Stanford dataset contains light fields whose view baseline and disparity range can be much larger. For this reason, the Stanford dataset does not closely resemble the typical data from a light field camera. However, we include the Stanford dataset in our experiments in order to evaluate the robustness of light field super-resolution methods to larger disparity ranges, possibly exceeding the assumed one. The Stanford light fields have all been acquired with a gantry, and they are characterized by a 17 × 17 array of views. Similarly to [29] and [32], in our experiments we crop the light fields to a 5 × 5 array of views, i.e., we choose \( M = 5 \).

In our experiments, we first create the low resolution version of the test light field from the datasets above. The spatial resolution of the test light field \( U \) is decreased by a factor \( \alpha \in \mathbb{N} \) by applying the blurring and sampling matrix \( SB \) of Eq. (2.4) to each color channel of each view. In order to match the assumptions of the other light field super-resolution frameworks involved in the comparison [29, 32], and without loss of generality, the blur kernel implemented by the matrix \( B \) is set to an \( \alpha \times \alpha \) box filter, and the matrix \( S \) performs a regular sampling. Then the obtained low resolution light field \( V \) is brought back to the original spatial resolution by the super-resolution algorithms under study. This approach guarantees that the final spatial resolution of the test light field is exactly its original one, regardless of \( \alpha \).

In our framework, the low resolution light field \( V \) is divided into possibly overlapping sub-light-fields and each one is reconstructed separately. Formally, a sub-light-field is obtained by fixing a spatial coordinate \((x, y)\) and then extracting an \( N' \times N' \) patch with the top left pixel at \((x, y)\) from each view \( V_{s,t} \). The result is an \( N' \times N' \times M \times M \) light field, with \( N' < (N/\alpha) \). Once all
Chapter 2. Geometrically consistent light field spatial super-resolution

The sub-light-fields are super resolved, different estimates of the same pixel produced by the possible overlap are merged. We set $N' = 100$ and 70 for $\alpha = 2$ and 3, respectively. This choice leads to a high resolution sub-light-field with a spatial resolution that is approximately $200 \times 200$ pixels. Finally, only the luminance channel of the low resolution light field is super resolved using our method, as the two chrominance channels can be easily up-sampled via bilinear interpolation due to their low frequency nature.

In our experiments we consider two variants of our Graph-Based super-resolution algorithm (GB). The first is GB-SQ, the main variant, which employs the warping matrix construction based on the square constraint (SQ) and is presented in Section 2.4.1. The second is GB-DR, the variant employing the warping matrices extracted directly (DR) from the graph and introduced at the end of Section 2.4.2. In the warping matrix construction in Eq. (2.10), as well as in the graph construction in Eq. (2.12), we empirically set the patch size $Q = 7$ pixels and $\sigma = 0.7229$. Throughout the thesis we assume the values of color and gray scale images to be in the range $[0,1]$. For the search window size, we set $C = 13$ pixels. This choice is equivalent to consider a disparity range of $[-6,6]$ pixels at high resolution. Note that this range happens to be loose for the HCI dataset, whose disparity range is within $[-3,3]$. Choosing exactly the correct disparity range for each light field could both improve the reconstruction by avoiding possible wrong correspondences in the graph and warping matrices, and decrease the computation time. On the other hand, for some light fields in the Stanford dataset, the chosen disparity range may become too small, thus preventing the possibility to capture the correct correspondences. In practice, the disparity range is not always available, hence the value $[-6,6]$ happens to be a fair choice considering that our super-resolution framework targets data from light field cameras, i.e., data with a typically small disparity range. Finally, after a grid search, we set $\lambda_2 = 0.2$ and $\lambda_g = 0.0055$ in the objective function in Eq. (2.3) and perform just two iterations of the full Algorithm 1, as we experimentally found that to be sufficient.

We carry out our experiments on the light field super-resolution algorithms in [29] using the original code by the authors, available online within the last release of the image processing library cocolib. For a fair comparison, we provide the $[-6,6]$ pixel range at the library input, in order to permit the removal of outliers in the estimated disparity maps. For our experiments on the algorithm in [32] we use the code provided by the authors. We discretize the $[-6,6]$ pixel range using a 0.2 pixel step, and for each disparity value we train a different GMM prior. The procedure is carried out for $\alpha = 2$ and 3, and results in GMM priors defined on a $4\alpha \times 4\alpha \times M \times M$ light field patch. A light field patch is equivalent to a sub-light-field, but with a very small spatial resolution. We perform the training on the data that comes together with the authors' code. We also compare GB with a state-of-the-art super-resolution method for single-frame super-resolution, and relying on a CNN [22]. For the comparison with the CNN-based super-resolution algorithm in [22], we employ the original code from the authors, available online. We perform the CNN training on the data provided together with the code. In particular, when generating the (low, high)-resolution patch pairs of the training set, we employ the blur and sampling matrix $SB$ of Eq. (2.4), where we recall that the blur kernel implemented by the matrix $B$ is set to an $\alpha \times \alpha$ box filter and the matrix $S$ performs a regular
2.5. Experiments

We learn one CNN for $\alpha = 2$ and one for $\alpha = 3$, and perform $8 \times 10^8$ back-propagations, as described in [22]. We then super-resolve each light field by applying the trained CNN on the single low resolution views. Finally, as a baseline reconstruction, we consider also the high resolution light field obtained from bilinear interpolation of the single low resolution views.

In the experiments, our super-resolution method GB, the methods in [32] and [22], and the bilinear interpolation one, super-resolve only the luminance of the low resolution light field. The full color high resolution light field is obtained through bilinear interpolation of the two low resolution light field chrominances. Instead, for the method in [29], the corresponding cocolib library needs to be fed with the full color low resolution light field and a full color high resolution light field is provided at the output. Since most of the considered super-resolution methods super-resolve only the luminance of the low resolution light field, we compute the reconstruction error only on the luminance channel. For the method in [29], whose light field luminance is not available directly, we compute it from the corresponding full color high resolution light field at the cocolib library output.

2.5.2 HCI and Stanford light field datasets

The numerical results from our super-resolution experiments on the HCI and Stanford datasets are reported in Tables 2.1 and 2.2 for a super-resolution factor $\alpha = 2$ and respectively for $\alpha = 3$ in Tables 2.3 and 2.4. For each reconstructed light field we compute the PSNR [dB] at each view and report the average and variance of the computed PSNRs in the tables. Finally, for a fair comparison with the method in [32], which suffers from border effects, a 15 pixel border is removed from all the reconstructed views before the PSNR computation.

Table 2.1 – HCI dataset [39]: PSNR mean and variance for the super-resolution factor $\alpha = 2$.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Bilinear</th>
<th>[29]</th>
<th>[32]</th>
<th>[22]</th>
<th>GB-DR</th>
<th>GB-SQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>buddha</td>
<td>35.22 ± 0.00</td>
<td>38.22 ± 0.11</td>
<td><strong>39.12 ± 0.62</strong></td>
<td>37.73 ± 0.03</td>
<td>38.59 ± 0.03</td>
<td>39.00 ± 0.14</td>
</tr>
<tr>
<td>buddha2</td>
<td>30.97 ± 0.00</td>
<td>33.01 ± 0.11</td>
<td>33.63 ± 0.22</td>
<td>33.67 ± 0.00</td>
<td>34.17 ± 0.01</td>
<td><strong>34.41 ± 0.02</strong></td>
</tr>
<tr>
<td>couple</td>
<td>25.52 ± 0.00</td>
<td>26.22 ± 1.61</td>
<td>31.83 ± 2.80</td>
<td>28.56 ± 0.00</td>
<td>32.79 ± 0.17</td>
<td><strong>33.51 ± 0.25</strong></td>
</tr>
<tr>
<td>cube</td>
<td>26.06 ± 0.00</td>
<td>26.40 ± 1.90</td>
<td>30.99 ± 3.02</td>
<td>28.81 ± 0.00</td>
<td>32.60 ± 0.23</td>
<td><strong>33.28 ± 0.35</strong></td>
</tr>
<tr>
<td>horses</td>
<td>26.37 ± 0.00</td>
<td>29.14 ± 0.63</td>
<td><strong>33.13 ± 0.72</strong></td>
<td>27.80 ± 0.00</td>
<td>30.99 ± 0.05</td>
<td>32.62 ± 0.12</td>
</tr>
<tr>
<td>maria</td>
<td>32.84 ± 0.00</td>
<td>35.60 ± 0.33</td>
<td>37.03 ± 0.44</td>
<td>35.50 ± 0.00</td>
<td>37.19 ± 0.03</td>
<td><strong>37.25 ± 0.02</strong></td>
</tr>
<tr>
<td>medieval</td>
<td>30.07 ± 0.00</td>
<td>30.53 ± 0.67</td>
<td>33.34 ± 0.71</td>
<td>31.23 ± 0.00</td>
<td>33.23 ± 0.03</td>
<td><strong>33.45 ± 0.02</strong></td>
</tr>
<tr>
<td>mona</td>
<td>35.11 ± 0.00</td>
<td>37.54 ± 0.64</td>
<td>38.32 ± 1.14</td>
<td>39.07 ± 0.00</td>
<td>39.30 ± 0.04</td>
<td><strong>39.37 ± 0.05</strong></td>
</tr>
<tr>
<td>papillon</td>
<td>36.19 ± 0.00</td>
<td>39.91 ± 0.15</td>
<td>40.59 ± 0.89</td>
<td>39.88 ± 0.00</td>
<td><strong>40.94 ± 0.06</strong></td>
<td>40.70 ± 0.04</td>
</tr>
<tr>
<td>pyramid</td>
<td>26.49 ± 0.00</td>
<td>26.73 ± 1.42</td>
<td>33.35 ± 4.06</td>
<td>29.69 ± 0.00</td>
<td>34.63 ± 0.34</td>
<td><strong>35.41 ± 0.67</strong></td>
</tr>
<tr>
<td>statue</td>
<td>26.32 ± 0.00</td>
<td>26.15 ± 2.15</td>
<td>32.95 ± 4.67</td>
<td>29.65 ± 0.00</td>
<td>34.81 ± 0.38</td>
<td><strong>35.61 ± 0.73</strong></td>
</tr>
<tr>
<td>stillLife</td>
<td>25.28 ± 0.00</td>
<td>25.58 ± 1.41</td>
<td>28.84 ± 0.82</td>
<td>27.27 ± 0.00</td>
<td>30.80 ± 0.07</td>
<td><strong>30.98 ± 0.05</strong></td>
</tr>
</tbody>
</table>

For a super-resolution factor $\alpha = 2$ in the HCI dataset, GB provides the highest average PSNR on ten out of twelve light fields. In particular, nine out of ten of the highest average PSNRs are due to GB-SQ. The highest average PSNR in the two remaining light fields buddha and horses is achieved by [32], but the corresponding variances are non negligible. The large variance generally indicates that the quality of the central views is higher than the one of the lateral
Chapter 2. Geometrically consistent light field spatial super-resolution

Table 2.3 – HCI dataset [39]: PSNR mean and variance for the super-resolution factor $\alpha = 3$.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Bilinear</th>
<th>[29]</th>
<th>[32]</th>
<th>[22]</th>
<th>GB-DR</th>
<th>GB-SQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>buddha</td>
<td>32.58 ± 0.01</td>
<td>34.92 ± 0.63</td>
<td>35.36 ± 0.34</td>
<td>34.62 ± 0.01</td>
<td>35.42 ± 0.02</td>
<td>35.40 ± 0.02</td>
</tr>
<tr>
<td>buddha2</td>
<td>28.14 ± 0.00</td>
<td>29.96 ± 0.07</td>
<td>30.29 ± 0.10</td>
<td>30.23 ± 0.00</td>
<td>30.52 ± 0.00</td>
<td>30.43 ± 0.00</td>
</tr>
<tr>
<td>couple</td>
<td>22.62 ± 0.00</td>
<td>23.02 ± 1.56</td>
<td>27.43 ± 1.16</td>
<td>24.01 ± 0.00</td>
<td>26.65 ± 0.01</td>
<td>26.95 ± 0.00</td>
</tr>
<tr>
<td>cube</td>
<td>23.25 ± 0.00</td>
<td>22.47 ± 2.65</td>
<td>26.48 ± 1.16</td>
<td>24.58 ± 0.00</td>
<td>27.23 ± 0.01</td>
<td>27.39 ± 0.00</td>
</tr>
<tr>
<td>horses</td>
<td>24.35 ± 0.00</td>
<td>24.45 ± 1.27</td>
<td>29.90 ± 0.55</td>
<td>24.73 ± 0.00</td>
<td>25.53 ± 0.00</td>
<td>26.41 ± 0.01</td>
</tr>
<tr>
<td>maria</td>
<td>30.02 ± 0.00</td>
<td>30.64 ± 0.67</td>
<td>33.36 ± 0.37</td>
<td>31.55 ± 0.00</td>
<td>33.46 ± 0.01</td>
<td>33.12 ± 0.01</td>
</tr>
<tr>
<td>medieval</td>
<td>28.29 ± 0.00</td>
<td>28.54 ± 0.17</td>
<td>29.78 ± 0.50</td>
<td>28.57 ± 0.00</td>
<td>29.23 ± 0.00</td>
<td>29.54 ± 0.01</td>
</tr>
<tr>
<td>mona</td>
<td>32.05 ± 0.00</td>
<td>33.42 ± 0.52</td>
<td>33.31 ± 0.40</td>
<td>34.82 ± 0.00</td>
<td>34.66 ± 0.01</td>
<td>34.47 ± 0.01</td>
</tr>
<tr>
<td>papillon</td>
<td>33.66 ± 0.00</td>
<td>36.76 ± 0.13</td>
<td>36.15 ± 0.48</td>
<td>36.56 ± 0.00</td>
<td>36.44 ± 0.01</td>
<td>36.18 ± 0.01</td>
</tr>
<tr>
<td>pyramide</td>
<td>23.39 ± 0.00</td>
<td>23.60 ± 2.72</td>
<td>29.13 ± 1.86</td>
<td>24.84 ± 0.00</td>
<td>28.34 ± 0.01</td>
<td>28.48 ± 0.00</td>
</tr>
<tr>
<td>statue</td>
<td>23.21 ± 0.00</td>
<td>22.97 ± 3.63</td>
<td>28.93 ± 2.03</td>
<td>24.72 ± 0.00</td>
<td>28.21 ± 0.01</td>
<td>28.38 ± 0.00</td>
</tr>
<tr>
<td>stillLife</td>
<td>23.28 ± 0.00</td>
<td>23.62 ± 1.64</td>
<td>27.23 ± 0.49</td>
<td>23.83 ± 0.00</td>
<td>24.99 ± 0.00</td>
<td>25.54 ± 0.00</td>
</tr>
</tbody>
</table>

Table 2.4 – Stanford dataset [36]: PSNR mean and variance for the super-resolution factor $\alpha = 3$.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Bilinear</th>
<th>[29]</th>
<th>[32]</th>
<th>[22]</th>
<th>GB-DR</th>
<th>GB-SQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>amethyst</td>
<td>32.69 ± 0.01</td>
<td>21.94 ± 0.30</td>
<td>31.47 ± 1.07</td>
<td>34.79 ± 0.01</td>
<td>35.97 ± 0.03</td>
<td>35.63 ± 0.02</td>
</tr>
<tr>
<td>beans</td>
<td>43.81 ± 0.01</td>
<td>22.66 ± 0.26</td>
<td>31.25 ± 1.85</td>
<td>47.38 ± 0.02</td>
<td>48.67 ± 0.12</td>
<td>47.28 ± 0.43</td>
</tr>
<tr>
<td>bracelet</td>
<td>29.06 ± 0.00</td>
<td>17.37 ± 0.22</td>
<td>15.83 ± 0.32</td>
<td>31.96 ± 0.00</td>
<td>33.23 ± 0.06</td>
<td>30.46 ± 1.08</td>
</tr>
<tr>
<td>bulldozer</td>
<td>31.88 ± 0.00</td>
<td>21.65 ± 0.11</td>
<td>26.21 ± 0.85</td>
<td>35.48 ± 0.01</td>
<td>35.44 ± 0.05</td>
<td>35.31 ± 0.04</td>
</tr>
<tr>
<td>bunny</td>
<td>39.03 ± 0.00</td>
<td>23.40 ± 1.22</td>
<td>35.88 ± 1.82</td>
<td>43.62 ± 0.01</td>
<td>43.73 ± 0.05</td>
<td>43.54 ± 0.05</td>
</tr>
<tr>
<td>cards</td>
<td>26.13 ± 0.00</td>
<td>17.77 ± 0.33</td>
<td>25.22 ± 1.40</td>
<td>28.34 ± 0.00</td>
<td>31.03 ± 0.02</td>
<td>31.49 ± 0.03</td>
</tr>
<tr>
<td>chess</td>
<td>33.11 ± 0.00</td>
<td>20.56 ± 0.24</td>
<td>31.19 ± 1.96</td>
<td>35.76 ± 0.00</td>
<td>36.87 ± 0.04</td>
<td>36.76 ± 0.03</td>
</tr>
<tr>
<td>eucalyptus</td>
<td>31.71 ± 0.00</td>
<td>23.38 ± 0.17</td>
<td>32.23 ± 1.61</td>
<td>33.03 ± 0.00</td>
<td>34.51 ± 0.01</td>
<td>34.80 ± 0.01</td>
</tr>
<tr>
<td>knights</td>
<td>31.31 ± 0.02</td>
<td>19.36 ± 0.07</td>
<td>25.55 ± 1.40</td>
<td>34.38 ± 0.00</td>
<td>35.37 ± 0.06</td>
<td>35.21 ± 0.05</td>
</tr>
<tr>
<td>treasure</td>
<td>27.98 ± 0.00</td>
<td>21.45 ± 0.14</td>
<td>27.86 ± 0.89</td>
<td>29.58 ± 0.00</td>
<td>31.37 ± 0.01</td>
<td>31.21 ± 0.01</td>
</tr>
<tr>
<td>truck</td>
<td>33.52 ± 0.02</td>
<td>23.27 ± 0.05</td>
<td>33.04 ± 1.66</td>
<td>35.45 ± 0.04</td>
<td>36.67 ± 0.05</td>
<td>36.97 ± 0.05</td>
</tr>
</tbody>
</table>

views. This is clearly non ideal, as our objective is to reconstruct all the views with high quality, as necessary in most light field applications. We also note that GB provides a better visual quality in these two light fields. This is shown in Figure 2.5 and 2.6, where two details from the bottom right-most views of the light fields buddha and horses, respectively, are given for each method. In particular, the reconstruction provided by [32] exhibits strong artifacts along object boundaries. This method assumes a constant disparity within each light field patch that it processes, but patches capturing object boundaries are characterized by an abrupt change.
2.5. Experiments

of disparity that violates this assumption and causes unpleasant artifacts. Figures 2.5c and 2.6c show that also the reconstructions provided by the method in [29] exhibit strong artifacts along edges, although the disparity is estimated at each pixel in this case. This is due to the presence of errors in the estimated disparity at object boundaries. These errors are caused both by the poor performance of the tensor structure operator in the presence of occlusions, and more in general to the challenges posed by disparity estimation at low resolution. We also observe that the TV term in [29] tends to over-smooth the fine details, as evident in the dice of Figure 2.5c. The method in [22], meant for single-frame super-resolution and therefore agnostic of the light field structure, provides PSNR values that are significantly lower than those provided by GB and [32], which instead take the light field structure into account. In particular, the quality of the views reconstructed by the method in [22] depends exclusively on the training data, as it does not employ the complementary information available at the other views. This is clear in Figure 2.5e, where [22] does not manage to recover the fine structure around the black spot in the dice, which remains pixelated as in the original low resolution view. Similarly, the method in [22] does not manage to reconstruct effectively the letters in Figure 2.6e, which remain blurred and in some cases cannot be discerned. Moreover, since the method in [22] does not consider the light field structure, it does not necessarily preserve it. An example is provided in Figure 2.7, where an epipolar image is extracted from the reconstructions of the stillLife light field computed by GB-SQ and the method in [22]. While GB-SQ preserves the line patterns, the method in [22] does not. The bilinear interpolation method provides the lowest PSNR values and the poor quality of its reconstruction is confirmed by the Figures 2.5b and 2.6b, which appear significantly blurred. In particular, the fine structure around the black spot in the dice of Figure 2.5h is almost absent in the reconstruction provided by the bilinear interpolation method, and some letters in Figure 2.6b cannot be discerned. Finally, the numerical results suggest that our GB-SQ methods is more effective in capturing the correct correspondences between adjacent views in the light field. A visual example is provided in Figure 2.8, where the letters in the view reconstructed by GB-SQ are sharper than those in the view reconstructed by GB-DR.

In the Stanford dataset and for the same super-resolution factor $\alpha = 2$, GB provides the highest average PSNRs on eight light fields out of eleven, the method in [22] provides the highest average PSNRs in the three remaining light fields, while the algorithms in [29] and [32] perform even worse than bilinear interpolation in most of the cases. The very poor performance of [29] and [32], and the generally higher PSNR provided by GB-DR compared to GB-SQ, are mainly consequences of the Stanford dataset disparity range, which exceeds the $[-6, 6]$ pixel range assumed in our tests. In particular, objects with a disparity outside the assumed disparity range are not properly reconstructed in general. An example is provided in Figure 2.9, where two details from the bottom right-most view of the light field bulldozer are shown. The detail at the bottom captures the bulldozer blade, placed very close to the camera and characterized by large disparity values outside the assumed disparity range, while the detail on the top captures a cylinder behind the blade and characterized by disparity values within the assumed range. As expected, GB manages to correctly reconstruct the cylinder, while it introduces some
Chapter 2. Geometrically consistent light field spatial super-resolution

![Figure 2.5](image)

Figure 2.5 – Detail from the bottom right-most view of the light field buddha, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with the method [29] in (c), with the method [32] in (d), with the method [22] in (e), with GB-DR in (f) and with GB-SQ in (g). The original high resolution light field is provided in (h).

artifacts on the blade. However, it can be observed that GB-DR introduces milder artifacts than GB-SQ on the blade, as GB-SQ forces the warping matrices to fulfill the square constraint of Section 2.4.1 on a wrong disparity range, while GB-DR is more accommodating in the warping matrix construction and therefore more robust to a wrong disparity range assumption. For the sake of completeness, Figure 2.9h provides the reconstruction computed by GB-SQ when the assumed disparity range is extended to $[-12, 12]$ pixels, and it shows that the artifacts disappear when the correct disparity range is within the assumed one. On the other hand, in Figure 2.9d the method in [32] fails to reconstruct also the cylinder, as the top of the image exhibits depth discontinuities that do not fit its assumption of constant disparity within each light field patch. The method in [29] fails in both areas as well, and in general on the whole Stanford light field dataset, as the structure tensor operator cannot detect large disparity values [40]. Differently from the light-field super-resolution methods, the one in [22] processes each view independently and it does not introduce any visible artifact, neither in the top nor in the bottom detail. However, the absence of visible artifacts does not guarantee that the light field structure is preserved, as [22] does not take it into account. For the sake of completeness, we observe that not all the light fields in the Stanford dataset meet the Lambertian assumption. Some areas of the captured scenes violate it. This contributes to the low PSNR values exhibited by the methods [29], [32], and GB-SQ, on certain light fields (e.g., bracelet) in Table 2.2, as in non Lambertian areas the light field structure in Eq. (2.2) does not hold true. On the other
2.5. Experiments

(a) LR  
(b) Bilinear  
(c) [29]  
(d) [32]  
(e) [22]  
(f) GB-DR  
(g) GB-SQ  
(h) Original HR

Figure 2.6 – Detail from the bottom right-most view of the light field horses, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with the method [29] in (c), with the method [32] in (d), with the method [22] in (e), with GB-DR in (f) and with GB-SQ in (g). The original high resolution light field is provided in (h).

In hand, as we already stated, GB-DR is more accommodating in the warping matrix construction and this makes the method more robust not only to the adoption of incorrect disparity ranges, but also to the violation of the Lambertian assumption, as confirmed numerically in Table 2.2.

We now consider a larger super-resolution factor of $\alpha = 3$. In the HCI dataset, the method in [32] provides the highest average PSNRs on half of the light fields, while GB provides the highest average PSNRs only on four of them. However, the average PSNR happens to be a very misleading index here. In particular, the method in [32] provides the highest average PSNR on the light field statue, but the PSNR variance is larger than 2 dB, which indicates a very large difference in the quality of the reconstructed images. On the other hand, GB-SQ provides a slightly lower average PSNR on the same light field, but the PSNR variance is 0.01 dB, which suggests a more homogenous quality of the reconstructed light field views. In particular, the lowest PNSR provided by GB-SQ among all the views is equal to 28.21 dB, which is almost 3 dB higher than the worst case view reconstructed by [32]. Moreover, the light fields reconstructed by [32] exhibit very strong artifacts along object boundaries. An example is provided in Figure 2.10, which represents a detail from the central view of the light field statue. The head of the statue reconstructed by [32] appears very noisy, especially at the depth discontinuity between the head and the background, while GB is not significantly affected. The lower average PSNR provided by GB on some light field, when compared to [32],
Figure 2.7 – Epipolar Plane Image (EPI) from the light field stillLife, in the HCI dataset [39]. The $9 \times 9$ light field is super-resolved by a factor $\alpha = 2$ using the single-frame super-resolution method in [22] and GB-SQ. The same EPI is extracted from the original HR light field, in (a), and from the reconstructions provided by [22] and GB-SQ, in (b) and (c), respectively. Since the method in [22] super-resolves the views independently, the original line pattern appears compromised, therefore the light field structure is not preserved. On the contrary, GB-SQ preserves the original line pattern, hence the light field structure.

is caused by the very poor resolution of the input data for $\alpha = 3$, that makes the capture of the correct matches for the warping matrix construction more and more challenging. However, as suggested by Figure 2.10, the regularizer $g(\cdot)$ manages to compensate for these errors. The method in [29] performs worse than [32] and GB both in terms of PSNR and visually. As an example, in Figure 2.10 the reconstruction provided by [29] shows strong artifacts not only at depth discontinuities, but especially in the background, which consists of a flat panel with a tree motive. Despite the very textured background, the tensor structure fails to capture the correct depth due to the very low resolution of the views, and this has a dramatic impact on the final reconstruction. In general, depth estimation at very low resolution happens to be a very challenging task. The method in [22] reconstructs the statue of Figure 2.10 correctly, and no unpleasant artifacts are visible. However, it introduces new structures in the textured background and this leads the PSNR to drop. In general, the method in [22] provides lower
2.5. Experiments

(a) GB-DR  
(b) GB-SQ

Figure 2.8 – Detail from the central view of the super-resolved light field horses, in the HCI dataset [39], for the super-resolution factor $\alpha = 2$. The reconstruction provided by GB-SQ exhibits sharper letters than the reconstruction by GB-DR, as the square constraint captures better the light field structure.

Figure 2.9 – Details from the bottom right-most view of the light field bulldozer, in the Stanford dataset [36]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with the method [29] in (c), with the method [32] in (d), with the method [22] in (e), with GB-DR in (f) and with GB-SQ in (g). The reconstruction of GB-SQ with the extended disparity range $[-12,12]$ pixels is provided in (h) and the original high resolution light field is in (i).

average PSNR values than GB on the twelve light fields, as the separate processing of each views makes it agnostic of the complementary information in the others and it can rely only on the data it scanned in the training phase. Finally, the worst numerical results are provided mainly by the bilinear interpolation method, which does not exhibit strong artifacts in general, but provides very blurred images, as shown in Figure 2.10b and expected.
Chapter 2. Geometrically consistent light field spatial super-resolution

Figure 2.10 – Detail from the bottom right-most view of the light field statue, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 3$ with bilinear interpolation in (b), with the method [29] in (c), with the method [32] in (d), with the method [22] in (e), with GB-DR in (f) and with GB-SQ in (g). The original high resolution light field is provided in (h).

Finally, for the Stanford dataset and $\alpha = 3$, the numerical results in Table 2.4 show a similar behavior to the one observed for $\alpha = 2$. The methods in [29] and [32] are heavily affected by artifacts, due to the disparities exceeding the assumed range. Instead, GB proves to be more robust to the incorrect disparity range, in particular the variant GB-DR. The method in [22] is limited by its considering the views separately, although it is not affected by the artifacts caused by the incorrect disparity range.

2.5.3 Lytro ILLUM dataset

We test our algorithm also on the MMSPG dataset [41], where real world scenes are captured with a hand held Lytro ILLUM camera [13]. The super-resolution task happens to be very challenging, as the views in each light field are characterized by a very low resolution and contain artifacts due to both the uncontrolled light conditions and the demosaicking process. Moreover, the Lambertian assumption is not always met. In the tests we keep the parameter setup described in Section 2.5.1, included the $[-6,6]$ disparity range. However, no PSNR is available as the light fields are directly super-resolved. In Figure 2.11 we provide five examples of light field views super-resolved by a factor $\alpha = 2$ with GB-SQ and the method in [32], as the latter represents GB’s main competitor in the tests of Section 2.5.2. Consistently with the previous experiments, at depth discontinuities the method in [32] leads to unpleasant artifacts,
while GB-SQ preserves the sharp transitions.

To conclude, our experiments over three datasets show that the proposed super-resolution algorithm GB has some remarkable reconstruction properties that make it preferable over its considered competitors. First, its reconstructed light fields exhibit a better visual quality, often confirmed numerically by the PSNR measure. In particular, GB leads to sharp edges while avoiding the unpleasant artifacts due to depth discontinuities. Second, it provides an homogeneous and consistent reconstruction of all the views in the light field, which is a fundamental requirement for light field applications. Third, it is more robust than the other considered methods in those scenarios where some objects in the scene exceed the assumed disparity range, as it may be the case in practice (e.g., in the MMSPG dataset), where there is no control on the scene.

2.6 Conclusions

We presented a new light field super-resolution algorithm, that exploits the complementary information encoded in the different views to augment their spatial resolution; the algorithm relies on a graph to regularize the overall light field. In particular, we showed that coupling an approximate warping matrix construction strategy with a graph regularizer, which enforces the light field geometric structure, avoids to carry out a costly disparity estimation step at sub-pixel precision. In addition, the proposed algorithm reduces to a quadratic problem, which can be solved efficiently with standard convex optimization tools.

The proposed algorithm compares favourably to the state-of-the-art light field super-resolution frameworks, both in terms of PSNR and visual quality. It provides a homogeneous reconstruction of all the views in the light field, which is a property that is not present in the other light field super-resolution frameworks [29] [32]. Also, although the proposed algorithm is meant mainly for light field camera data, where the disparity range is typically small, it is flexible enough to handle light fields with larger disparity ranges too. We also compared our algorithm to a state-of-the-art single-frame super-resolution method based on CNNs [22], and we showed that taking the light field structure into account allows our algorithm to recover finer details, and most importantly it avoids the reconstruction of a set of geometrically inconsistent high resolution views.

Finally, since the proposed super-resolution algorithm relies only on the complementary information in the low resolution light field views, the amount of details that it can introduce decreases when the super-resolution factor $\alpha$ increases, faster than for learning based-methods. In fact, learning-based super-resolution methods can be trained for a specific super-resolution factor, therefore they can implicitly rely on external data at inference time, rather than on the sole complementary information in the views. This is clearly a limitation of the proposed method. A promising direction could be represented by a combination of the two approaches, in order to permit higher super-resolution factors while preserving the light field structure.
Chapter 2. Geometrically consistent light field spatial super-resolution

Figure 2.11 – Details from the central view of the light field Bikes (first and second row), Chain_link_Fence_2 (third row), Flowers (fourth row), and Fountain_&_Vincent (fifth row) from the MMSPG dataset [41]. The original low resolution images in the first column are super-resolved by a factor $\alpha = 2$ with the methods [32] and GB-SQ in the second and third columns, respectively.
In Chapter 2 we addressed the light field super-resolution problem adopting a quadratic regularizer, specifically the first quadratic form of the graph Laplacian, in Eq. (2.8). Although everywhere differentiable, quadratic regularizers are known to induce a low-pass filtered solution, which may not be desirable when the we aim at recovering sharp edges and fine structures in general. Even if quadratic graph-based regularizers are less biased toward smooth solutions than traditional quadratic regularizers, e.g., the Tikhonov regularizer [42], it is worth analyzing the benefits of adopting a nonsmooth graph-based regularization.

In this chapter we analyze a nonsmooth graph-based regularizer, the Non Local Total Variation (NLTV) introduced in Eq. (1.6). The chapter is organized as follows. In Section 3.1 we compare NLTV to the quadratic graph-based regularizer in Eq. (2.8) and explain the benefits of the first one over the latter. Then, in Sections 3.2 and 3.3 we consider NLTV in the regularization of two specific inverse problems. In particular, Section 3.2 builds over Chapter 2 and compares the light field super-resolution framework when employing the quadratic graph-based regularizer and NLTV. Section 3.3 extends this regularizer to another task, and explores the use of NLTV in the context of omnidirectional stereo: given two omnidirectional cameras in a scene, the disparity of one of them must be estimated to reconstruct the scene geometry. Finally, Section 3.4 concludes the chapter.

### 3.1 Non Local Total Variation

The quadratic graph-based regularizer adopted in Chapter 2 reads as follows:

\[ g(u) = \sum_{i} \sum_{j \sim i} W(i, j) (u(j) - u(i))^2, \]  

(3.1)

where each term of the double sum \( W(i, j)(u(j) - u(i))^2 \) requires the constraint \( u(i) = u(j) \) to be fulfilled more or less tightly depending on the weight \( W(i, j) \). The main weakness of the

---

1Part of the work presented in this chapter has been published in [43].
Chapter 3. Nonsmooth super-resolution and omnidirectional stereo

The regularizer in Eq. (3.1) can be stressed out by rewriting it as follows:

\[ g(u) = \sum_{(i,j) \in \mathcal{E}} W(i,j) (u(j) - u(i))^2, \]  

(3.2)

where we recall that \( \mathcal{E} \) is the set of edges of the graph \( \mathcal{G} \). Eq. (3.2) makes clear that the regularizer aggregates all the terms \( W(i,j) (u(j) - u(i))^2 \) regardless of their affiliation to a specific neighborhood \( \{ j \sim i \} \). As a consequence, while minimizing the regularizer, each term is minimized independently of the others and results in each constraint \( u(i) = u(j) \) being enforced independently. However, in the context of light field super-resolution, the neighborhood \( \{ j \sim i \} \) of the pixel \( i \) comprises all those pixels \( j \) belonging to the same epipolar line of \( i \), which we recall is formed by the projections of the same 3D point of the scene into the different light field views as shown in Figure 2.3b, therefore we would expect the constraint \( u(i) = u(j) \) to be fulfilled for all the pixels \( j \in \{ j \sim i \} \) jointly.

Let us now focus on the NLTV regularizer [9], which reads as follows:

\[ g(u) = \sum_{i} \sqrt{\sum_{j \sim i} W(i,j)} (u(j) - u(i))^2. \]  

(3.3)

At a first sight, the NLTV regularizer looks very similar to its quadratic counterpart in Eq. (3.1). In order to outline the main differences between both regularizers, we can rewrite the NLTV regularizer in Eq. (3.3) as follows:

\[ g(u) = \sum_{i} \left\| \frac{\sqrt{W(i,j_1)} (u(j_1) - u(i))}{\sqrt{W(i,j_2)} (u(j_2) - u(i)) \ldots \sqrt{W(i,j_K)} (u(j_K) - u(i))} \right\|_2. \]

(3.4)

where \( \{ j_1, j_2, \ldots, j_K \} = \{ j \sim i \} \). The \( \ell_1 \)-norm is known to promote sparse vectors, therefore the minimization of the NLTV regularizer in Eq. (3.4) would try to put to zero each entry \( q_i \) of the vector \( q \) independently. Zeroing an entry \( q_i \) translates into fulfilling the constraint \( u(i) = u(j) \) for each \( j \in \{ j \sim i \} \) jointly, which is the desired behavior that is missing in the quadratic regularizer in Eq. (3.1). We finally observe that \( q_i = \nabla_q u(i) \) is the gradient of \( u \) at the pixel \( i \) on the graph \( \mathcal{G} \), hence the NLTV regularizer generalizes the TV regularizer of Eq. (1.4) to the graph setting.

To summarize, in Eq. (3.4) the \( \ell_2 \)-norm is first applied to each vector \( q_i \) and the resulting vector \( q \) in Eq. (3.4) is aggregated using an \( \ell_1 \)-norm. The result is a mixed norm referred to as
3.2 Nonsmooth light field super-resolution

\[ 3.2. \text{Nonsmooth light field super-resolution} \]

\[ \ell_{1,2}-\text{norm}. \text{Thanks to this norm, NLTV enforces all the terms } W(i, j)(u(j) - u(i))^2 \text{ associated} \]

\[ \text{to the same neighborhood } (j \sim i) \text{ to be minimized jointly. This property, typically referred to} \]

\[ \text{as group sparsity, makes NLTV a more effective regularizer than its quadratic counterpart, as it} \]

\[ \text{leads to better edge preserving properties in general.} \]

3.2 Nonsmooth light field super-resolution

In this section we test the use of the NLTV regularizer in the context of light field super-resolution: this will permit to analyze the benefits of NLTV over the quadratic regularizer in Eq. (3.1). Similarly to Chapter 2, we cast the super-resolution problem into the minimization of the following objective function:

\[ u^* \in \arg\min_u f_1(u) + \lambda_2 f_2(u) + \lambda g(u) \]  

(3.5)

where \( f_1(\cdot) \) is the data fidelity term defined in Equations (2.5), \( f_2(\cdot) \) is the warping term defined in Equations (2.7) while \( g(\cdot) \) is chosen to be the NLTV regularizer.

3.2.1 Super-resolution algorithm

We consider the minimization of the objective function in Eq. (3.5), which is still convex, but now it involves a nonsmooth term \( g(\cdot) \). We introduce the linear operator \( \mathcal{T} \), which maps the vector \( u \) to a set of \((NM)^2\) other vectors; it is defined as follows:

\[ \mathcal{T}(u) = \{q_1, \ldots, q_i, \ldots, q_{(NM)^2}\}. \]

Now the NLTV regularizer \( g(\cdot) \) can be expressed as the composition of the linear operator \( \mathcal{T} \) with the mixed \( \ell_{1,2} \)-norm:

\[ g(u) = \| \mathcal{T}(u) \|_{1,2} = \left\| \left\| q_1 \|_2, \| q_2 \|_2, \ldots, \| q_{(NM)^2} \|_2 \right\|_1 \right. \]

The convex objective function in Eq. (3.5) involves the sum of a differentiable function \( f(\cdot) = f_1(\cdot) + \lambda_2 f_2(\cdot) \) and a term \( g(\cdot) \) which is the composition of a linear operator with a nonsmooth function. In convex optimization, the state-of-art solvers for this class of problems are represented by Primal-Dual Proximal Methods [44–49]. Among the possible solvers, we choose the Forward-Backward Primal-Dual (FBPD) method [48], due to its straightforward implementation. The FBPD method is detailed in Algorithm 2, where \( \| \cdot \|_{op} \) is the operator norm and \( \mathcal{T}^* \) is the adjoint operator of \( \mathcal{T} \). Each iteration evaluates the gradient \( \nabla f \) of the function \( f(\cdot) = f_1(\cdot) + \lambda_2 f_2(\cdot) \) and the proximity operator of the \( \ell_{1,2} \)-norm. The gradient \( \nabla f \) is defined as follows:

\[ \nabla f(u) = 2 \sum_k (SB)^T (SBu_k - v_k) \]
\[ + 2 \sum_{k} \sum_{k' \in \mathcal{K}_k^+} \lambda_2 \left( SBF_k^T \right) \left( SBF_k^T u_k' - v_k \right). \]

The \( \ell_{1,2} \)-norm proximity operator \([50]\) evaluated at a set of vectors \( \{q_1, \ldots, q_i, \ldots, q_{(NM)^2}\} \) is again a set of vectors:

\[
\text{prox}_{\|\cdot\|_{1,2}} \left( \{q_1, \ldots, q_i, \ldots, q_{(NM)^2}\} \right) = \{p_1, \ldots, p_i, \ldots, p_{(MN)^2}\}
\]

where

\[
p_i = \begin{cases} 
0 & \text{if} \quad \|q_i\|_2 \leq \mu, \\
\left(1 - \frac{\mu}{\|q_i\|_2}\right) q_i & \text{otherwise}.
\end{cases}
\]

We conclude by observing that Algorithm 2 can simply replace the CG solver in Algorithm 1 to solve the light field super-resolution problem presented in Chapter 2.

### 3.2.2 Experiments

In this section we present some numerical and visual examples of the tests of our super-resolution algorithm with graph-based nonsmooth regularizer, called GB-NS hereafter, on the HCI light field dataset \([39]\) introduced in Section 2.5. We compare GB-NS to our algorithm GB-SQ and to the light field super-resolution method in \([32]\). Also in these experiments, we provide the results of a simple bilinear interpolation of the single views as a baseline.

For a fair comparison between the methods GB-NS and GB-SQ, in our experiments GB-NS adopts the same parameters used to build the warping matrices and the graph in Section 2.5. Regarding the weights in Eq.\((3.5)\), we set \(\lambda_2 = 0.15\) and \(\lambda_g = 0.0055\) after a grid search. Similarly to Section 2.5, we solve the optimization problem in Eq. \((3.5)\) twice: at the first round the warping matrices and the graph are built on a bilinearly interpolated version of the light field views, then the obtained high resolution light field is used to build the warping matrices and the graph for the second round. Finally, for the method in \([32]\), we adopt the same model trained in Section 2.5.

---

**Algorithm 2** FBPD \([48]\)

**Input:** \(v = [v_1, \ldots, v_{M^2}], u^{[0]}, \text{iter}\)

**Output:** \(u = [u_1, \ldots, u_{M^2}]\)

1. \(z^{[0]} \leftarrow \mathcal{F}(u^{[0]})\) \quad \text{▷ Initialization}
2. Choose \(\tau, \omega \in \mathbb{R} \geq 0\) such that \(\tau (\beta/2 + \omega \lambda_g \|\mathcal{T}\|_{op}^2) < 1\) \quad \text{▷ Initialization}
3. for \(i = 1\) to \text{iter} do
   4. \(\hat{u}^{[i]} \leftarrow \nabla f(u^{[i]}) + \mathcal{T}^* (\hat{z}^{[i]})\) \quad \text{▷ Primal update}
   5. \(u^{[i+1]} \leftarrow u^{[i]} - \tau \hat{u}^{[i]}\) \quad \text{▷ Primal update}
   6. \(\hat{z}^{[i]} \leftarrow \mathcal{T}(2u^{[i+1]} - u^{[i]})\) \quad \text{Dual update}
   7. \(z^{[i+1]} \leftarrow \left(\hat{z}^{[i]} + \omega \hat{z}^{[i]}\right) - \omega \text{prox}_{\frac{\|\cdot\|_{1,2}}{\omega}} \left(\frac{\hat{z}^{[i]} + \omega \hat{z}^{[i]}}{\omega}\right)\) \quad \text{Dual update}
8. end for
9. return \(u\)

---

For a fair comparison between the methods GB-NS and GB-SQ, in our experiments GB-NS adopts the same parameters used to build the warping matrices and the graph in Section 2.5. Similarly to Section 2.5, we solve the optimization problem in Eq. \((3.5)\) twice: at the first round the warping matrices and the graph are built on a bilinearly interpolated version of the light field views, then the obtained high resolution light field is used to build the warping matrices and the graph for the second round. Finally, for the method in \([32]\), we adopt the same model trained in Section 2.5.
Since the main flavour of this chapter is to provide some insights on the effectiveness of nonsmooth regularization, adopted later on in this thesis, we report only the results of our experiment for the super-resolution factor \( \alpha = 2 \). For each light field in the HCI dataset, Table 3.1 reports the PSNR [dB] average and variance of the \( M^2 = 25 \) reconstructed views. The proposed method GB-NS achieves the highest PSNR on five out of twelve light fields. A visual example of the better reconstructions provided by GB-NS in these light fields is provided in the Figures 3.1 and 3.2, which show two details from the light fields mona and papillon, respectively. It can be observed that GB-NS provides sharper edges than GB-SQ close to object transitions. In Figure 3.1, this can be observed at the transition between the plant, in green and brown, and the background with the flower pattern. In Figure 3.2, the same phenomenon can be observed at the transition between the black butterfly wing and the green leaf in the background. In the light fields buddha and stillLife, GB-NS is no more than 0.03 dB away from the highest PSNR values, achieved by [32] and GB-SQ, respectively. Finally, concerning the remaining five light fields couple, cube, maria, pyramid and statue, GB-NS still achieves better visual results than its competitors [32] and GB-SQ. This can be observed in the Figures 3.3 and 3.4, which represent two details from the light fields pyramid and statue, respectively: GB-NS provides sharper edges than GB-SQ and the method [32], whose reconstructions look more blurred and pixilated. In fact, the higher PSNR exhibited by GB-SQ in the five light fields just mentioned is due to their particular structure: the foreground hosts an object that changes from light field to light field, while the background consists of the same panel with a fined detailed tree motif. At low resolution the panel details are completely lost, hence building a meaningful graph in the panel area is really challenging, both for GB-NS and GB-SQ. In fact, no significant visual difference can be perceived between the panels reconstructed by GB-NS and GB-SQ in the background area of Figures 3.3 and 3.4. However, thanks to the smooth nature of its regularizer, GB-SQ reconstructs a smoother panel texture where the error is spread over the whole surface. On the other hand, in each one of the five considered light fields, the object in the foreground exhibits significantly sharper edges in the reconstruction by GB-NS than in those reconstructed by GB-SQ and [32], as shown again in the Figures 3.3 and 3.4.

Table 3.1 – HCI dataset [39]: PSNR mean and variance for the super-resolution factor \( \alpha = 2 \).

<table>
<thead>
<tr>
<th>Scene</th>
<th>Bilinear</th>
<th>[32]</th>
<th>GB-SQ</th>
<th>GB-NS</th>
</tr>
</thead>
<tbody>
<tr>
<td>buddha</td>
<td>35.22 ± 0.00</td>
<td><strong>39.12 ± 0.62</strong></td>
<td>39.00 ± 0.14</td>
<td>39.09 ± 0.10</td>
</tr>
<tr>
<td>buddha2</td>
<td>30.97 ± 0.00</td>
<td>33.63 ± 0.22</td>
<td>34.41 ± 0.02</td>
<td><strong>34.54 ± 0.01</strong></td>
</tr>
<tr>
<td>couple</td>
<td>25.52 ± 0.00</td>
<td>31.83 ± 2.80</td>
<td><strong>33.51 ± 0.25</strong></td>
<td>33.43 ± 0.24</td>
</tr>
<tr>
<td>cube</td>
<td>26.06 ± 0.00</td>
<td>30.99 ± 3.02</td>
<td><strong>33.28 ± 0.35</strong></td>
<td>33.11 ± 0.38</td>
</tr>
<tr>
<td>horses</td>
<td>26.37 ± 0.00</td>
<td>33.13 ± 0.72</td>
<td>32.62 ± 0.12</td>
<td><strong>33.59 ± 0.19</strong></td>
</tr>
<tr>
<td>maria</td>
<td>32.84 ± 0.00</td>
<td>37.03 ± 0.44</td>
<td><strong>37.25 ± 0.02</strong></td>
<td>37.02 ± 0.01</td>
</tr>
<tr>
<td>medieval</td>
<td>30.07 ± 0.00</td>
<td>33.34 ± 0.71</td>
<td>33.45 ± 0.02</td>
<td><strong>33.50 ± 0.01</strong></td>
</tr>
<tr>
<td>mona</td>
<td>35.11 ± 0.00</td>
<td>36.32 ± 1.14</td>
<td>39.37 ± 0.05</td>
<td><strong>40.05 ± 0.01</strong></td>
</tr>
<tr>
<td>papillon</td>
<td>36.19 ± 0.00</td>
<td>40.59 ± 0.89</td>
<td>40.70 ± 0.04</td>
<td><strong>41.36 ± 0.02</strong></td>
</tr>
<tr>
<td>pyramid</td>
<td>26.49 ± 0.00</td>
<td>33.35 ± 4.06</td>
<td><strong>35.41 ± 0.67</strong></td>
<td>35.09 ± 0.51</td>
</tr>
<tr>
<td>statue</td>
<td>26.32 ± 0.00</td>
<td>32.95 ± 4.67</td>
<td><strong>35.61 ± 0.73</strong></td>
<td>35.43 ± 0.62</td>
</tr>
<tr>
<td>stillLife</td>
<td>25.28 ± 0.00</td>
<td>28.84 ± 0.82</td>
<td><strong>30.98 ± 0.05</strong></td>
<td>30.96 ± 0.05</td>
</tr>
</tbody>
</table>
Figure 3.1 – Detail from the bottom right-most view of the light field mona, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with GB-NS in (c), with the method [32] in (d) and with GB-SQ in (e). The original high resolution light field is provided in (f).
Figure 3.2 – Detail from the bottom right-most view of the light field papillon, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with GB-NS in (c), with the method [32] in (d) and with GB-SQ in (e). The original high resolution light field is provided in (f).
Figure 3.3 – Detail from the bottom right-most view of the light field pyramid, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with GB-NS in (c), with the method [32] in (d) and with GB-SQ in (e). The original high resolution light field is provided in (f).
3.2. Nonsmooth light field super-resolution

Figure 3.4 – Detail from the bottom right-most view of the light field statue, in the HCI dataset [39]. The low resolution light field in (a) is super-resolved by a factor $\alpha = 2$ with bilinear interpolation in (b), with GB-NS in (c), with the method [32] in (d) and with GB-SQ in (e). The original high resolution light field is provided in (f).
3.3 Omnidirectional stereo

In this section we investigate the use of the NLTV regularizer in the context of omnidirectional stereo, i.e., the estimation of the scene geometry from a pair of omnidirectional cameras. Differently from perspective cameras, which are characterized by a limited field of view, an omnidirectional camera can capture the whole surrounding scene in a single exposure [51]. This particular feature has multiple applications. As an example, a user equipped with a head mounted display can enjoy an omnidirectional video by moving their head around, thus experiencing a deep immersion in the scene.

The omnidirectional stereo problem has recently gained new attention both from the industry and the research community. Originally omnidirectional cameras were introduced mainly to equip robots with a 360 field of view: the surrounding geometry was estimated from consecutive frames in order to provide the robot with a better understanding of the scene during its autonomous navigation. Today the advent of Virtual Reality (VR) makes omnidirectional stereo suitable to reconstruct real world scenarios and enable the user to navigate them. In fact, with data captured with two or more omnidirectional cameras we could enable 6-Degrees-of-Freedom (6-DoF) navigation, where a user would not only be able to move their head around, but could potentially walk in the captured scene [52]. Although 6-DoF could be addressed also using a large number of perspective cameras, omnidirectional cameras represent a more natural tool for this kind of application. Therefore, since geometry estimation from omnidirectional cameras is a key problem to enable 6-DoF, in this section we focus on omnidirectional stereo. Some preliminary work exist in this domain already [53, 54], but they do not take into account the particular structure of the omnidirectional data.

Multiple omnidirectional camera systems are compliant with the spherical unified model [55]. This model permits to treat an omnidirectional camera as a sphere, with the sphere surface being the camera image plane and the center of the sphere being the camera center of projection. Within this model, an omnidirectional image is a signal living on the sphere. Traditional image regularizers tailored for signal living on a 2D Euclidean domain are not ideal to process images living on a sphere. On the other hand, graph-based regularizer can be used to model arbitrary topologies, therefore they can be used to process omnidirectional images directly on the sphere. Based on this observation, in this section we propose to extend the NLTV regularizer to model omnidirectional data and address the problem of depth map estimation in a setup involving two omnidirectional cameras, hence a stereo setup.

3.3.1 Problem formulation

In accordance with the unified model in [55], we model an omnidirectional camera as a sphere $S \subset \mathbb{R}^3$ of unitary radius, as represented in Figure 3.5. The $(X, Y, Z)$ camera coordinate system has its origin at the center of $S$ and a point $P = (X, Y, Z) \in \mathbb{R}^3$ in the scene is projected onto the sphere surface at $P/\|P\|_2 = (X_S, Y_S, Z_S) \in S$. A point on the sphere $S$ can be represented using the polar coordinates $(\phi, \theta)$, with $\phi \in [0, 2\pi]$ and $\theta \in [0, \pi]$ the azimuthal and elevation
3.3. Omnidirectional stereo

Figure 3.5 – Omnidirectional stereo model. The two omnidirectional cameras are modeled as two unitary spheres. In particular, the reference system \{\bar{X}, \bar{Y}, \bar{Z}\} of the bottom camera is obtained by translating the reference system \{X, Y, Z\} of the top camera along the negative semi-axis of Z. The point \(P\) is imaged at the coordinates \((\phi, \theta)\) and \((\phi, \bar{\theta})\) on the surfaces of the top and bottom cameras, respectively.

Coordinates, respectively. In particular, we assume the following sphere parametrization:

\[
\chi: (\phi, \theta) \rightarrow \begin{bmatrix} X_S \\ Y_S \\ Z_S \end{bmatrix} = \begin{bmatrix} -\sin \phi \sin \theta \\ -\cos \phi \sin \theta \\ \cos \theta \end{bmatrix}. \tag{3.6}
\]

Finally, we assume that the sphere is sampled according to an equiangular grid, obtained by sampling both the \(\phi\) and the \(\theta\) axes with a step size \(\Delta \phi = \Delta \theta = \pi / N\). Within this widely used sampling scheme, an omnidirectional image \(I\) can be represented as an \(N \times 2N\) tensor where the pixel \((x, y)\) with \(x = 0, 1, \ldots, 2N - 1\) and \(y = 0, 1, \ldots, N - 1\) corresponds to the polar coordinates \((\phi, \theta)\) with \(\phi = x \Delta \phi + (\Delta \phi / 2)\) and \(\theta = y \Delta \theta + (\Delta \theta / 2)\).

We consider the two cameras as vertically registered [56] and estimate the disparity rather than the depth map, as these are just equivalent representations of the scene geometry. If we denote with \(\{X, Y, Z\}\) the coordinate system of the top camera, assumed to be the reference one, the coordinate system \(\{\bar{X}, \bar{Y}, \bar{Z}\}\) of the bottom camera is obtained by translating \(\{X, Y, Z\}\) along the negative semi-axis of \(Z\) by the chosen baseline \(b \in \mathbb{R}_{>0}\). A point \(P\) imaged in the reference
Chapter 3. Nonsmooth super-resolution and omnidirectional stereo

designed to enforce similar disparity values at those pixels which are considered to be strongly correlated.

Let \( I^t \in \mathbb{R}^{N \times 2N} \) and \( I^b \in \mathbb{R}^{N \times 2N} \) be the images captured by the top and bottom omnidirectional cameras, respectively. We cast the omnidirectional disparity estimation problem into the minimization of the following cost function:

\[
D^* = \arg\min_D \ f_1(D) + \lambda_2 f_2(D) + \lambda_g g(D), \tag{3.7}
\]

where \( f_1(\cdot) \) and \( f_2(\cdot) \) are data terms enforcing photo consistency between the two omnidirectional images and \( g(\cdot) \) is the NLTV regularizer. The constants \( \lambda_2 \) and \( \lambda_g \in \mathbb{R}_{\geq 0} \) balance the three terms. The data term \( f_1(\cdot) \) reads as follows:

\[
f_1(D) = \sum_{(x,y)} \min \left\{ \left| I^t(x,y) - I^b(x,y + D(x,y)) \right|, \tau_1 \right\}
\]

with \( \tau_1 \in \mathbb{R}_{>0} \). It enforces photo consistency in the plain intensity domain. Instead, the data term \( f_2(\cdot) \) is defined as follows:

\[
f_2(D) = \sum_{(x,y)} \min \left\{ \| \nabla_S I^t(x,y) - \nabla_S I^b(x,y + D(x,y)) \|_2, \tau_2 \right\}
\]

with \( \tau_2 \in \mathbb{R}_{>0} \). It enforces photo consistency in the intensity gradient domain. It is important to note that, since the images \( I^t \) and \( I^b \) are defined on the sphere, \( \nabla_S(\cdot) \) is the surface gradient operator for the sphere [57]. Specifically, the surface gradient \( \nabla_S I^t(x,y) = \nabla_S I^t(\phi,\theta) \in \mathbb{R}^3 \) is defined as follows:

\[
\nabla_S I^t(\phi,\theta) = \nabla \Phi(\phi,\theta) (G(\phi,\theta))^{-1} \nabla I^t(\phi,\theta),
\]

where \( \nabla I^t(\phi,\theta) \in \mathbb{R}^2 \) is the spatial gradient in \( \mathbb{R}^2 \) and \( G(\phi,\theta) = (\nabla \Phi(\phi,\theta))^\top \nabla \Phi(\phi,\theta) \in \mathbb{R}^{2 \times 2} \) with \( \nabla \Phi(\phi,\theta) \in \mathbb{R}^{3 \times 2} \) is the Jacobian of the sphere parametrization at \( (\phi,\theta) \). Finally, for the sake of completeness, we report also the NLTV regularizer \( g(\cdot) \):

\[
g(D) = \sum_i \sqrt{\sum_{j \neq i} W(i,j) (D(j) - D(i))^2}, \tag{3.8}
\]

where the index \( i \) represents the linear index associated to the pixel coordinates \( (x,y) \), equivalently \( (\phi,\theta) \). The graph adjacency matrix \( W \) is designed to capture the omnidirectional disparity map correlation directly on the sphere. In particular, as described below, the graph is designed to enforce similar disparity values at those pixels which are considered to be strongly correlated.
3.3. Omnidirectional stereo

3.3.2 Disparity estimation algorithm

The key part of the omnidirectional stereo algorithm is represented by the graph construction, which shall capture the particular structure of the omnidirectional disparity map. In fact, due to the equiangular sampling, the pixel density increases when moving from the sphere equator toward the poles, therefore assuming the same correlation model everywhere in the $N \times 2N$ omnidirectional image would not be ideal.

We proceed by introducing the geodesic distance between two coordinates $(\phi, \theta)$ and $(\bar{\phi}, \bar{\theta})$ on the unitary sphere:

$$d_{geo}((\phi, \theta), (\bar{\phi}, \bar{\theta})) = 2 \arcsin \left( \frac{d_{eu}((\phi, \theta), (\bar{\phi}, \bar{\theta}))}{2} \right),$$

(3.9)

where $d_{eu}(\cdot)$ denotes the Euclidean distance in $\mathbb{R}^3$. Then, for each pixel $(\phi, \theta)$ in the equiangular grid, we establish an edge in the graph between the pixel itself and each other pixel at coordinates $(\bar{\phi}, \bar{\theta})$ such that $d_{geo}((\phi, \theta), (\bar{\phi}, \bar{\theta})) \leq R$, where $R \in \mathbb{R}_{>0}$. This approach guarantees that the neighbourhood of each pixel $(\phi, \theta)$ covers the same surface area on the sphere. In fact, due to the denser sampling of the equiangular pattern towards the poles of the sphere, defining a squared neighborhood directly in the $N \times 2N$ equiangular grid would lead the connections of the pixels towards the poles to be more and more local, which would result in an unfair treatment of the different areas of the sphere.

Each graph edge connecting two pixels is weighted according to both their geodesic distance on the sphere and their intensity similarity in the reference camera. In particular, the weight of the edge between the pixels $(\phi, \theta)$ and $(\bar{\phi}, \bar{\theta})$, with linear indexes $i$ and $j$, respectively, is defined as follows:

$$W(i, j) = \exp \left( - \frac{d_{geo}((\phi, \theta), (\bar{\phi}, \bar{\theta}))}{\sigma_{geo}^2} \right) \exp \left( - \frac{\|Q(\phi, \theta) - Q(\bar{\phi}, \bar{\theta})\|_F^2}{\sigma_{int}^2} \right),$$

(3.10)

where $Q(i) \in \mathbb{R}^{Q \times Q}$ is a square patch centered at the coordinates $(\phi, \theta)$ but defined on the plane tangent to the sphere at $(\phi, \theta)$, $\| \cdot \|_F$ denotes the Frobenius norm, $\sigma_{int}$ and $\sigma_{geo} \in \mathbb{R}_{>0}$ are tunable parameters. The patches $Q(i)$ and $Q(j)$ are extracted from the tangent planes at the coordinates $(\phi, \theta)$ and $(\bar{\phi}, \bar{\theta})$, respectively, in order to compare patches covering the same surface area on the sphere. In fact, a $Q \times Q$ patch defined on the equiangular grid is characterized by a surface area on the sphere $S$ which shrinks as the center coordinate $(\phi, \theta)$ of the patch moves toward the poles of the sphere. Examples of tangent patches are illustrated in Figure 3.6. The first exponential in Eq. (3.10) makes the weight decay as the geodesic distance between $(\phi, \theta)$ and $(\bar{\phi}, \bar{\theta})$ increases: in fact, it is reasonable to assume that the closer two pixels are on the sphere the more they are correlated, therefore the more likely they exhibit similar disparities. On the other hand, the second exponential provides a high weight when the patches centered at two pixels are similar, and a low one otherwise: this avoids strong
connections between two pixels that may correspond to different objects in the scene.

Figure 3.6 – Example of tangent patches.

Altogether, the optimization problem in Eq. (3.7) is no longer convex, due to the term \( f(\cdot) \). We decide to solve it using a Sub-Gradient Descent with momentum, in particular we resort to ADAM [58], due to both its straightforward implementation and the fast convergence, timewise, observed for the resolution of the omnidirectional images adopted in our experiments.

3.3.3 Experiments

We create a small omnidirectional stereo dataset containing three stereo pairs, each one obtained by rendering a different 3D scene using Blender [59]. This approach permits the creation of omnidirectional images fulfilling the assumed spherical model and equipped with the corresponding ground truth depth maps, which can be converted to disparity for our experiments. We rendered three scenes: office and castle, whose Blender models are borrowed from a web platform for 3D model sharing, and city [62]. For office, which is an indoor scene, we adopt a 20 cm baseline. For city, which is an outdoor scene but whose content is not far away from the camera, we adopt an 80 cm baseline. Finally, for castle, which is also an outdoor scene but whose content is far away from the camera, we adopt a much larger baselines equal to 8 m.

We compare the proposed stereo algorithm with the one obtained by replacing our geometry aware regularizer \( g(\cdot) \) in Eq. (3.8) with the NE-like regularizer in Eq. (1.5), which ignores the spherical geometry. We will refer to this stereo algorithm as NE hereafter. Both our algorithm and NE are initialized with the disparity map obtained via Semi-Global Matching (SGM) [63] on the omnidirectional stereo pair, which is also agnostic of the spherical geometry. In particular, we employ the SGM implementation provided in OpenCV [64] with default parameters.

Regarding our algorithm, we empirically set the graph connection radius \( R = 2 \Delta \theta \), the patch size \( Q = 3 \) pixels and the weight tunable parameters \( \sigma_{geo} = R/3 \) and \( \sigma_{int} = 0.2 \). After a grid search over both the indoor and outdoor scenes, we set \( \lambda_2 = \lambda_g = 1 \) and \( \tau_1 = \tau_2 = 0.2 \). Regarding the NE algorithm instead, after a grid search, we set the tensor parameters \( \gamma = 0.5 \)

\[ \text{2The Blender files for the scenes office and castle are available at [60] and [61], respectively.} \]
3.4 Conclusions

and $\beta = 1$ and we set $\lambda_2 = \lambda_g = 1$ and $\tau_1 = \tau_2 = 0.2$; the gradients are computed using a $5 \times 5$ pixel Gaussian derivative filter with standard deviation equal to 0.2 pixels.

Table 3.2 – Omnidirectional stereo. The table compares the disparity maps computed by SGM [63], NE and our algorithm, using the bad pixel metric. For each scene and metric, the best error is in bold.

<table>
<thead>
<tr>
<th>Scene</th>
<th>1px</th>
<th>2px</th>
<th>3px</th>
<th>4px</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SGM</td>
<td>NE</td>
<td>Ours</td>
<td>SGM</td>
</tr>
<tr>
<td>castle</td>
<td>41.70</td>
<td>35.81</td>
<td>30.61</td>
<td>33.41</td>
</tr>
<tr>
<td>city</td>
<td>11.84</td>
<td>10.10</td>
<td>8.81</td>
<td>9.09</td>
</tr>
</tbody>
</table>

We evaluate SGM, NE and our algorithm using the bad px metric, which captures the number of pixels with a disparity error larger than a given threshold. Table 3.2 reports the results of our experiments when 1, 2, 3 and 4 pixels are considered as thresholds. It can be observed that, except for the bad 1px metric in the office scene, our algorithm provides always the lowest error. The estimated disparity maps, together with the corresponding bad 2px error masks, are reported in the Figures 3.7, 3.8 and 3.9 for the scenes office, castle and city, respectively. The improvement of our algorithm over NE and SGM can be appreciated by observing that many yellow speckles, which denote areas with error larger than 2 pixels, are present in the NE and especially in the SGM error maps, but disappear in our method ones. This is particularly evident both on the table and on the walls in the office scene and on the areas of the floor nearby the walls in the castle scene. Overall, the presented results suggest that adopting a graph-based regularizer, which takes into account the spherical topology underneath omnidirectional images, permits to better handle the distortion caused by the equiangular sampling and to better estimate the scene geometry.

We conclude by observing that the proposed omnidirectional stereo algorithm is meant to be mainly a proof of concept for the flexibility and the effectiveness of graph-based regularizers, in particular nonsmooth ones, in the regularization of problems where the target signal has an arbitrary complex structure. In fact, the pixel wise data terms $f_1(\cdot)$ and $f_2(\cdot)$ are not very discriminative and can lead to ambiguous matchings, especially in untextured areas. However, despite this clear weakness, the graph-based regularizer manages to significantly improve the input SGM disparity map.

3.4 Conclusions

In this chapter we analyzed NLTV, an example of nonsmooth graph-based regularizer. We started by observing that NLTV, thanks to its mixed $\ell_{1,2}$-norm, enforces that all the constraints associated to the same pixel in the graph are fulfilled jointly, which is a particularly desirable and expected property when adopting a graph-based regularizer. Then, we analyzed NLTV in the context of both light field super-resolution and omnidirectional stereo.

As far as light field super-resolution is concerned, we carried out a direct comparison between
Chapter 3. Nonsmooth super-resolution and omnidirectional stereo

Figure 3.7 – Omnidirectional stereo on the office scene. The first row hosts the reference image and the ground truth disparity, on the first and second column, respectively. Each other row hosts the bad 2px disparity error mask and the corresponding disparity map, on the first and second column, respectively. In particular, the rows from two to three refer to the stereo methods SGM [63], NE and ours, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
3.4. Conclusions

Figure 3.8 – Omnidirectional stereo on the castle scene. The first row hosts the reference image and the ground truth disparity, on the first and second column, respectively. Each other row hosts the bad 2px disparity error mask and the corresponding disparity map, on the first and second column, respectively. In particular, the rows from two to three refer to the stereo methods SGM [63], NE and ours, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
Figure 3.9 – Omnidirectional stereo on the city scene. The first row hosts the reference image and the ground truth disparity, on the first and second column, respectively. Each other row hosts the bad 2px disparity error mask and the corresponding disparity map, on the first and second column, respectively. In particular, the rows from two to three refer to the stereo methods SGM [63], NE and ours, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
the quadratic graph-based regularizer presented in Chapter 2 and NLTV. In particular, we showed that replacing the quadratic regularizer with NLTV in our super-resolution framework permits to recover sharper edges and finer details, which are both crucial properties in super-resolution. We also showed, as a proof concept, that NLTV can be successfully extended to regularize an omnidirectional disparity map within an omnidirectional stereo problem. Although the traditional TV [2] has been widely used to regularize perspective disparity maps, it cannot model the correlation of a signal living on a sphere correctly. Instead, thanks to the graph underneath, NLTV is flexible enough to adapt to arbitrary surface, in particular the sphere. The flexibility of graph-based regularizers and effectiveness of the nonsmooth ones will be stressed out further in the next chapter, where a new nonsmooth graph-based regularizer will be introduced in the context of depth refinement and normal estimation for perspective cameras.
Joint depth refinement and normal estimation with a planar bias

In the attempt to create intelligent agents capable of interacting with the 3D world around us, dense depth estimation represents a crucial problem. Typically, this is approached either using active devices, like Time-Of-Flight cameras, or via dense stereo matching methods [63, 65–68], which rely on two or more images of the same scene to compute its geometry. Active methods suffer from noisy measurements, possibly caused by light interference or multiple reflections, therefore they can benefit from a post-processing step refining the measured depth map. Similarly, dense stereo matching methods tend to fall short in untextured areas, where the matching becomes ambiguous, or in the presence of occlusions, therefore, typically a stereo matching pipeline includes a refinement step to fill the missing depth map areas and remove the noise.

In general, the refinement step is guided by the image associated to the measured or estimated depth map. Most of the depth refinement literature focuses on enforcing some kind of first order smoothness among the depth map pixels, possibly avoiding smoothing across the edges of the guide image, as these may correspond to object, hence depth, transitions [69–71]. Although depth maps are definitely piece-wise smooth, first order smoothness is a very general assumptions, which does not exploit the geometrical simplicity of most 3D scenes. Based on the observation that most human made environments are characterized by planar surfaces, some authors propose to enforce second order smoothness by computing a set of possible planar surfaces a priori and assigning each depth map pixel to one of them [72]. Unfortunately, this refinement strategy imposes to select a finite number of plane candidates a priori, which may not be optimal. In fact, first the complexity of the scene geometry could vary significantly from scene to scene; second, fixing the planes a priori could potentially bias the depth refinement toward incorrect planes, as the set of candidate planes is the object of an additional estimation procedure.

In this chapter\textsuperscript{1} we propose a depth map refinement framework which promotes piece-wise planar scenes, leveraging the fact that most human made environments are characterized by

\textsuperscript{1}The work presented in this chapter is covered in the articles [73, 74], currently under review.
planar surfaces, but which does not require any a priori knowledge of the planar surfaces in the scene. We cast the depth refinement problem into the minimization of a cost function involving two main terms: a data term penalizes those solutions deviating from the input depth map in areas whose depth is considered as reliable, a regularization term promotes, explicitly, depth maps corresponding to piece-wise planar surfaces in the 3D scene. In particular, our regularizer models the depth map as a weighted graph, built on the guide image, where the nodes are the pixels and where the strength of the connection between two pixels captures the probability that their corresponding points in the 3D scene belong to the same planar surface.

Our contribution is twofold. On the one hand, we propose a graph-based depth refinement framework introducing a novel regularizer which promotes the reconstruction of piece-wise planar scenes explicitly, but, thanks to the graph underneath, it is flexible enough to handle non fully piece-wise planar scenes as well. On the other hand, since our regularizer models the scene explicitly as piece-wise planar, our depth refinement framework estimates, jointly, a normal map of the scene as well.

Our depth refinement and normal estimation framework is potentially very useful in the context of large scale 3D reconstruction, where the large number of images to process requires fast dense stereo matching methods which rely on a later refinement, and where the fusion of the depth maps in a single point cloud can benefit from the estimated normals [74]. We test our framework extensively and show that our framework is effective in both refining the input depth map and estimating the corresponding normal map.

The chapter is organized as follows. Section 4.1 provides an overview of the existing literature on depth map refinement. Section 4.2 motivates the novel regularization term and derives the geometry underneath it. Section 4.3 presents our problem formulation and Section 4.4 presents our full algorithm. In Section 4.5 we carry out extensive experiments to test the effectiveness of the proposed depth refinement and normal estimation framework. Finally, in Section 4.6 we wrap up our proposed method and provide some hints on possible future improvements.

4.1 Related work

Depth refinement methods fall mainly into three classes: local methods [70, 75, 76], global methods [69, 72] and learning-based methods [77, 78].

Local methods are characterized by a greedy approach: each pixel in the depth map, or disparity map, is refined sequentially without taking any global consistency into account. As an example, Tosi et al. [70] adopt a two step strategy. First, the input disparity map is used to compute a binary confidence mask that classifies each pixel as reliable or not. Then, the disparity at the pixels classified as reliable is kept unchanged and used to infer the disparity at the non reliable ones, using a wise interpolation heuristic. In particular, for each non reliable pixel, a set of anchor pixels with a reliable disparity is selected and the pixel disparity is estimated
as a weighted average of the anchor disparities. Besides its low computational requirements, the method in [70] suffers two major drawbacks. On the one hand, pixels classified as reliable are left unchanged: this does not permit to correct possible pixels misclassified as reliable and, in addition, these outliers may bias the refinement of those pixel correctly classified as non reliable, thus impacting the framework robustness to outliers. On the other hand, the method in [70], and local methods in general, cannot take advantage of the reliable parts of the disparity map fully, due to their local perspective.

Global methods rely on an optimization procedure to refine each pixel of the input disparity map jointly. Barron and Poole [69] propose the Fast Bilateral Solver, a framework which permits to cast arbitrary image related ill posed problems into a global optimization formulation, whose regularizer resembles the popular bilateral filter [79]. In [70] the Fast Bilateral Filter has been shown to be effective in the disparity refinement task, but its general purposefulness prevents it from competing with specialized methods, even local ones like [70].

Global is also the disparity refinement framework proposed by Park et al. [72], which can be broken down into four steps. First, the input reference image is partitioned into super-pixels and a local plane is estimated for each one of them using RANSAC. Second, super-pixels are progressively merged into macro super-pixels to cover larger areas of the scene and a new global plane is estimated for each of them. Then, a Markov Random Field (MRF) is defined over the set of super-pixels and each one is assigned to one of four different classes: the class associated the local plane of the super-pixel, the class associated to the global plane of the macro super-pixels to which the super-pixel belongs, the class of pixels not belonging to any planar surface, or the class of outliers. The MRF employs a regularizer that enforces connected super-pixel to belong to the same class, thus promoting a global consistency of the disparity map. Finally, the parameters of the plane associated to each super-pixel are slightly perturbed, again within a MRF model, to allow for a finer disparity refinement. This method is the closest to ours in flavour. However, the a priori detection of a finite number of candidate planes for the whole scene biases the refinements toward a set of plane hypothesis that may either not be correct, as estimated on the input noisy and possibly incomplete disparity map, or not be rich enough to cover the full geometry of the scene.

Finally, recent learning based methods typically rely on a deep neural network which, fed with the noisy or incomplete disparity map, outputs a refined version of it [77, 78]. In [77] the task is split into three sub-tasks, each one addressed by a different network and finally trained end to end as a single one: detection of the non reliable pixels, gross refinement of the disparity map, and fine refinement. Instead, Knöbelreiter and Pock [78] revisit the work of Cherabier et al. [80] in the context of disparity refinement. First, the disparity refinement task is cast into the minimization of a cost function, hence a global optimization, whose minimizer is the desired refined disparity map. However, the cost function is partially parametrized, rather than fully handcrafted. Then, the cost function solver can be unrolled for a fixed number of iterations, thus obtaining a network structure, and the parametrized cost function can be learned. Once the network parameters are learned, the disparity refinement requires just a
network evaluation. Both the methods in [77] and [78] permit a fast refinement of the input disparity. However, due to their learning-based nature, they can falls short easily in those scenarios which differ from the ones employed at training time, as shown for the method in [78], which performs remarkably well in the Middlebury benchmark [81] training set, while quite poorly in the test set of the same dataset.

Our graph-based depth refinement framework, instead, does not rely on any training procedure. It adopts a global approach which permits to compute, jointly, a pair of consistent depth and normal maps. Moreover, it does not need any a priori knowledge of the possible planar surfaces in the scene, as it automatically assigns a plane to each pixel based on its neighbors in the underneath graph. Finally, the proposed framework does not call for a separate handling of pixels belonging to planar surfaces and not, again thanks to the graph underneath.

4.2 Depth map model

In this section we investigate the relation between a plane in the 3D space and its 2D depth map, assuming a Pinhole camera model. In Subsection 4.2.1 we show that, when a plane in the 3D space is imaged by a camera, the corresponding 2D inverse depth map is described by a plane as well, thus motivating a piece-wise planar model for the inverse depth map of those scenes where planar structures are prevalent. In Section 4.2.2, instead, we proceed backward and show how to recover the original plane in the 3D space, in particular its normal, when the inverse depth map of the imaged plane is available.

4.2.1 Plane projection

Let us consider a plane \( \mathcal{P} \) in the 3D scene in front of a pinhole camera. The plane \( \mathcal{P} \) can be uniquely described by a pair \((P_0, n_0)\), where \( P_0 = (X_0, Y_0, Z_0) \in \mathcal{P} \subset \mathbb{R}^3 \) is a point of the plane and \( n_0 = (a_0, b_0, c_0) \in \mathbb{R}^3 \), with \( ||n_0||_2 = 1 \), is a vector defining the plane orientation, referred to as the plane normal. The equation of the plane reads as follows:

\[
\langle n_0, (X, Y, Z) - (X_0, Y_0, Z_0) \rangle = 0,
\]

where \( P = (X, Y, Z) \) is a generic point of the plane and \( \langle \cdot, \cdot \rangle \) denotes the inner product operator. We can rewrite the previous equation by making its constant part explicit:

\[
\langle n_0, (X, Y, Z) \rangle + \langle n_0, -(X_0, Y_0, Z_0) \rangle = 0, \quad (4.1)
\]

where \( \rho_0 \in \mathbb{R} \) is a constant. Finally, remembering that is \( n_0 = (a_0, b_0, c_0) \), Eq. (4.1) can be rewritten as follows:

\[
a_0X + b_0Y + c_0Z - \rho_0 = 0. \quad (4.2)
\]
4.2. Depth map model

Eq. (4.2) is expressed in the left handed coordinate system of the pinhole camera, whose $Z$ axis points outside the camera and is aligned with the camera optical axis.

In the considered pinhole model, where the pixel coordinate origin $(0,0)$ is at the top left corner of the image, the 3D point $P = (X, Y, Z)$ is projected into the camera image plane at the image coordinates $(x, y) \in \mathbb{R}^2$:

$$
\begin{align*}
    x &= \frac{X}{Z} f_x + c_x, \\
    y &= \frac{Y}{Z} f_y + c_y,
\end{align*}
$$

(4.3)

where $(c_x, c_y) \in \mathbb{R}^2$ are the coordinates of the camera center of projection and $f_x, f_y$ are the horizontal and vertical focal lengths, respectively. We solve Eqs. (4.3) for $X$ and $Y$:

$$
\begin{align*}
    X &= \left(\frac{x - c_x}{f_x}\right) Z, \\
    Y &= \left(\frac{y - c_y}{f_y}\right) Z.
\end{align*}
$$

(4.4)

Replacing the Eqs. (4.4) in Eq. (4.2) provides the 3D plane equation as a function of the image coordinates $(x, y)$ and the depth $Z$:

$$
\left( a_0 \left(\frac{x - c_x}{f_x}\right) + b_0 \left(\frac{y - c_y}{f_y}\right) + c_0 \right) Z - \rho_0 = 0.
$$

(4.5)

We also observe that, by applying the Eqs. (4.4) to the expression of $\rho_0$ in Eq. (4.1), the constant $\rho_0$ can be expressed as follows:

$$
\rho_0 = \left( a_0 \left(\frac{x_0 - c_x}{f_x}\right) + b_0 \left(\frac{y_0 - c_y}{f_y}\right) + c_0 \right) Z_0,
$$

(4.6)

where $(x_0, y_0)$ is the projection of the point $P_0 = (X_0, Y_0, Z_0)$ into the camera image plane.

Remark. The pairs $(P_0, n_0)$ and $(P_0, -n_0)$ identify the same plane $\mathcal{P}$. However, if the plane represents a physical surface, e.g., a wall, then the two normals $n_0$ and $-n_0$ can be thought as representing the two sides of the plane. We choose to represent the plane with the normal $n_0$ associated to the visible side of the plane. This is equivalent to require that the angle between $n_0$ and the vector from the point $P_0$ to the pinhole camera origin is acute. Formally, this translates into the following constraint:

$$
\langle n_0, -(X_0, Y_0, Z_0) \rangle > 0.
$$

(4.7)

which is equivalent to the constraint $\rho_0 > 0$ (see Eq. (4.1)).

Let us now express Eq. (4.5) using the inverse depth $d(x, y) = 1/Z(x, y)$, where we stress that $Z(x, y)$ and $d(x, y)$ are the depth and inverse depth, respectively, associated to the image.
We conclude with the following remark, which permits to establish a relation between the inverse depth:

\[
d(x, y) = \frac{1}{\rho_0} \left( \frac{a_0 (x - c^x)}{f_x} + \frac{b_0 (y - c^y)}{f_y} + c_0 \right)
= \left( \frac{a_0}{\rho_0 f_x} \right) x + \left( \frac{b_0}{\rho_0 f_y} \right) y + \frac{1}{\rho_0} \left( c_0 - \frac{a_0 c^x}{f_x} - \frac{b_0 c^y}{f_y} \right).
\] (4.8)

Now let us introduce the vector \( u(x_0, y_0) = (u^x_0, u^y_0) \in \mathbb{R}^2 \) defined as follows:

\[
u^x_0 = \frac{a_0}{\rho_0 f_x}, \quad u^y_0 = \frac{b_0}{\rho_0 f_y}.
\] (4.9)

Eq. (4.9) allows us to express Eq. (4.8) as follows:

\[
d(x, y) = d(x_0, y_0) + \langle u(x_0, y_0), (x - x_0, y - y_0) \rangle,
\] (4.10)

which shows that the inverse depth \( d(x, y) \) of every point \( P \in \mathcal{P} \) is described by a plane: in particular, this plane passes through the point \((x_0, y_0, d(x_0, y_0))\) and has a normal \((u(x_0, y_0), -1) \in \mathbb{R}^3 \). We refer to Appendix B.1 for the proof of the equivalence between Eq. 4.8 and Eq. (4.10).

To summarize, we showed that the plane \( \mathcal{P} \) is represented either by the pair \((P_0, n_0)\) in the scene domain or by the pair \((u(x_0, y_0, d(x_0, y_0)), u(x_0, y_0))\) in the camera domain. Equivalently, the inverse depth map of a planar surface imaged by the camera is planar as well.

### 4.2.2 Normal computation

Eq. 4.9 permits to derive \( u(x_0, y_0) = (u^x_0, u^y_0) \) when the normal \( n_0 = (a_0, b_0, c_0) \) is given. We are now interested in the inverse operation. When \((u^x_0, u^y_0)\) is given, adding the constrain \( \|n_0\|_2 = 1 \) to the equations in Eq. (4.9) leads to a nonlinear system in \( a_0, b_0, \) and \( c_0 \), whose solution permits to recover \( n_0 = (a_0, b_0, c_0) \):

\[
\begin{align*}
u^x_0 &= \left( \rho_0 f_x \right)^{-1} a_0 & (4.11a) \\
u^y_0 &= \left( \rho_0 f_y \right)^{-1} b_0 & (4.11b) \\
a_0^2 + b_0^2 + c_0^2 &= 1. & (4.11c)
\end{align*}
\]

The close form solution of the above system is provided in Appendix B.2.

We conclude with the following remark, which permits to establish a relation between the inverse depth \( d(x_0, y_0) \) and the normal \( n_0 = (a_0, b_0, c_0) \).

**Remark.** Eq. (4.10) can be interpreted as the first order Taylor expansion of the inverse depth \( d(x, y) \) at \((x_0, y_0)\), with \( u(x_0, y_0) = \nabla d(x_0, y_0) \):

\[
d(x, y) = d(x_0, y_0) + \langle \nabla d(x_0, y_0), (x - x_0, y - y_0) \rangle
= d(x_0, y_0) + \left( \frac{\partial d}{\partial x} (x_0, y_0), \frac{\partial d}{\partial y} (x_0, y_0) \right), (x - x_0, y - y_0)).
\] (4.12)
Remark 4.2.2 suggests that, if the depth map \( Z(x, y) \) of the scene is available, ideally, the corresponding normal map \( n(x, y) \) can be computed by first estimating the field \( u(x, y) \) as the spatial gradient of the inverse depth \( d(x, y) = 1/Z(x, y) \) and then converting \( u(x, y) \) into the normal map \( n(x, y) \) using Eq. (4.9). In practice, the gradient can be obtained by convolving \( d(x, y) \) with a Gaussian derivative filter, whose standard deviation can be set according to the level of noise that characterizes the depth map \( Z(x, y) \). On the one hand, a large standard deviation is more robust to possible noise in the depth map \( Z(x, y) \), but could lead to normals that capture the surface orientations only approximately. On the other hand, a small standard deviation permits to capture fine changes in the normal orientation, but it would also make the final normal map very sensible to noise in the depth map \( Z(x, y) \). The outlined normal estimation represents a useful tool when an accurate depth map is available, as a small standard deviation can be used and fine changes in the plane orientations can be captured. However, regardless of the quality of the depth map, the filter standard deviation should tend to zero in order to provide accurate normals at the edges in the depth map: in practice, this is not possible, due to the discretization of the filter, and normals cannot be recovered correctly at depth edges. Our depth refinement and normal estimation framework does not suffer this limitation instead.

### 4.3 Problem formulation

Urban and indoor scenes are the main subject of depth estimation procedures. These scenes are characterized by a prevalence of human made structures, which, in most of the cases, are either inherently piece-wise planar or can be well approximated as such. Based on this observation and on the fact that the pinhole camera model maps a planar surface in the scene into a planar inverse depth map, as showed in Section 4.2, we propose to refine directly the inverse input depth map by enforcing it to be piece-wise planar. This is achieved by casting the inverse depth map refinement task into the minimization of a cost function comprising two main terms: a data term penalizes those solutions deviating from the input inverse depth map where this is considered as reliable, a regularization term promotes piece-wise planar solutions explicitly. In particular, the regularizer models the inverse depth map as a weighted graph, where the pixels are the nodes, and where strongly connected pixels in the graph are enforced to belong to the same plane. The regularizer is designed to estimate a plane automatically at each pixel, without any need for an \textit{a priori} estimation of the scene main planes, and at the same time is designed to enforce that strongly connected pixels are assigned to the same plane. In the following, in Section 4.3.1 we present our cost function in detail and show how the normal map is estimated jointly with the refined inverse depth map, then in Section 4.3.2 we compare the proposed regularizer to its closest counterpart in the literature, in order to outline its benefits.
4.3.1 Depth map refinement

In the following, the tensors $Z \in \mathbb{R}^{N \times N}$ and $N \in \mathbb{R}^{N \times N \times 3}$ represent the discretized versions of the depth map $Z(x, y)$ and the normal map $n(x, y)$, respectively, introduced in the previous Section 4.2. Similarly, $D \in \mathbb{R}^{N \times N}$ and $U \in \mathbb{R}^{N \times N \times 2}$ represent the discretized versions of the quantities $d(x, y)$ and $u(x, y)$, respectively.

Given an image $I \in \mathbb{R}^{N \times N}$, we are interested in recovering the corresponding depth map $Z$ when only a noisy and possibly incomplete estimate $\tilde{Z}$ is available. We assume that $\tilde{Z}$ is provided together with a confidence mask $M \in \mathbb{R}^{N \times N}$ with entries in $[0,1]$. In particular, when $i = (x, y) \in \{0, 1, \ldots, N\} \times \{0, 1, \ldots, N\}$ indicates the 2D pixel coordinates, the confidence map is such that $M(i) = 0$ when the entry $\tilde{Z}(i)$ is considered completely inaccurate, while $M(i) = 1$ when $\tilde{Z}(i)$ is considered highly accurate.

In the following, we focus on estimating the refined inverse depth map $D = 1/Z$ given $\tilde{D} = 1/\tilde{Z}$. In particular, we cast the inverse depth map refinement task into the following optimization problem:

$$D^* \in \mathop{\arg\min}_{D, U} f(D) + \lambda g(D, U)$$

(4.13)

where $f(\cdot)$ is a data term for the inverse depth map $D$ and $g(\cdot)$ is our proposed graph-based regularizer for piece-wise planar functions, with $w$ the graph weights. The scalar $\lambda \in \mathbb{R}_{\geq 0}$ balances the two terms of the cost function. The refined depth map is simply $Z^* = 1/D^*$.

The data fidelity term $f(\cdot)$ enforces that the estimated inverse depth map $D$ is close to $\tilde{D}$ at those pixels $i$ where the latter is considered accurate, i.e., where $M(i)$ tends to one:

$$f(D) = \sum_i \left| D(i) - \tilde{D}(i) \right| M(i).$$

(4.14)

In particular, this term will prevent the regularizer $g(\cdot)$ from attempting to refine those estimates $\tilde{D}(i)$ that are already considered as accurate.

The regularizer $g(\cdot)$ enforces that the inverse depth map $D$ is piece-wise planar. In particular, the regularizer models the inverse depth map as a weighted directed graph, where each pixel represents a node, and where the weight of the edge between two pixels can be interpreted as the likelihood that the corresponding two points in the 3D scene belong to the same plane. The regularizer parametrizes the inverse depth at each pixel with a different plane, but it enforces strongly connected pixels in the graph, i.e., those pixels connected by an edge with high weight, to share the same plane parametrization. Formally, the regularizer $g(\cdot)$ encompasses two

---

2A wide variety of algorithms addressing pixel-wise confidence prediction exist in the literature, either based on hand-crafted features or learning-based [82]. In practice, also the simple stereo re-projection error could be adopted [78].

3The quality of the confidence map $M$ can affect the quality of the refined depth map. However, in the case of missing confidence, i.e., $M$ constant, our formulation in Eq. (4.13) still promotes piece-wise planar scenes.
which requires the inverse depth map in the neighborhood of the pixel $i$, where $j_1, j_2, \ldots, j_{K_i}$ are the $K_i \in \mathbb{N}$ pixels directly connected to $i$ in the graph, i.e., pixel $i$ neighborhood, while $w(i, j) \in \mathbb{R}_{>0}$ with $j \in \{j_1, j_2, \ldots, j_{K_i}\}$ is the weight associated to the edge from the pixel $i$ to the neighboring pixel $j$. The term of the regularizer in Eq. (4.15) enforces the following constraint between the pixel $i$ and its neighboring pixel $j$:

$$\mathbf{D}(j) = \mathbf{D}(i) - \langle \mathbf{U}(i), j - i \rangle, \quad j \in \{j_1, j_2, \ldots, j_{K_i}\},$$  

(4.17)

which requires the inverse depth map in the neighborhood of the pixel $i$ to be approximated by a plane whose orientation is given by the vector $\mathbf{U}(i) \in \mathbb{R}^2$. This constraint recalls Eq. (4.10) and it is made more, or less, tight depending on the weight $w(i, j)$. On the other hand, the term of the regularizer in Eq. (4.16) can be interpreted as a generalization of the anisotropic Total Variation (TV) regularizer to the graph setting\(^4\), which leads to a regularizer which recalls the NLTVC presented in Section 3.1, and it still promotes piece-wise constant signals on the graph. According to Eq. (4.17), the vector $\mathbf{U}(i)$ defines the orientation of the plane describing the signal $\mathbf{D}$ in the neighborhood of the pixel $i$: enforcing the signal $\mathbf{U}$ to be piece-wise constant translates into enforcing the signal $\mathbf{D}$ to be piece-wise planar. Finally, the use of the graph weights in $g(\cdot)$ is crucial, as the scene is not fully piece-wise planar in practice. Therefore, indicated with $P_j$ and $P_i \in \mathbb{R}^3$ the points in the scene projected to the pixels $j$ and $i$, respectively, it is important to enforce the constraint in Eq. (4.17) more when the likelihood that the point $P_j$ belongs to the same plane of $P_i$ is high, less otherwise. The graph weights, as

\(^4\)Eq. (1.3) provides the original TV regularizer introduced in [2] and it is referred to as isotropic, due to the employed $\ell_2$-norm inside the sum. Instead, the TV version employing the $\ell_1$-norm inside the sum is referred to as the anisotropic version. It is important to note that, overall, the isotropic TV employs an $\ell_{1,2}$-norm; its generalization to the graph setting leads directly to the NLTVC [5] regularizer in Eq. (3.3), which was also observed to employ an $\ell_{1,2}$-norm. Instead, overall the anisotropic TV still employs an $\ell_1$-norm; its generalization to the graph setting leads to a regularizer which still employs an $\ell_1$-norm. All these regularizers promote piece-wise constant signals, either on the plane or on the graph. Whether using the isotropic or the anisotropic version depends on the specific application. In both cases, a further generalization, as in Eq. (4.16), is necessary when the signal to regularize is a vector field rather than a scalar one.
explained later in Section (4.4.2), are designed to capture this aspect.

We conclude by recalling that it is straightforward to convert each vector $\mathbf{U}(i) \in \mathbb{R}^2$ into the 3D normal $\mathbf{N}(i) \in \mathbb{R}^3$ by solving the non linear system in Eq. (4.11), whose solution is provided in close form in Appendix B.2.

### 4.3.2 Discussion

Our regularizer in Eqs. (4.15) and (4.16) resembles the well known *Non Local Total Generalized Variation (NLTGV)* [10], which reads as follows:

$$
g_{\text{NLTGV}}(\mathbf{D}, \mathbf{U}) = \sum_{i} \left[ w(i, j_1) \left| \mathbf{D}(j_1) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_1 - i \rangle \right| + w(i, j_2) \left| \mathbf{D}(j_2) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_2 - i \rangle \right| + \cdots + w(i, j_K) \left| \mathbf{D}(j_K) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_K - i \rangle \right| \right] + \alpha \sum_{i} \left( \left| \mathbf{U}^x(j_1) - \mathbf{U}^x(i) \right| + \left| \mathbf{U}^y(j_1) - \mathbf{U}^y(i) \right| \right) + \cdots + \left( \left| \mathbf{U}^x(j_K) - \mathbf{U}^x(i) \right| + \left| \mathbf{U}^y(j_K) - \mathbf{U}^y(i) \right| \right)$$

(4.18)

(4.19)

Let us start by comparing the first term of NLTGV and of our regularizer, in Eqs. (4.18) and (4.15), respectively. The two terms differ in the norm used to aggregate the entries of the vector $\mathbf{q}(i)$: the NLTGV regularizer employs an $\ell_1$-norm, while ours employs an $\ell_2$. The use of the $\ell_1$-norm in the NLTGV term in Eq. (4.18) permits to rewrite it as follows:

$$
\sum_{i} \sum_{k=1}^{K_i} w(i, j_k) \left| \mathbf{D}(j_k) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_k - i \rangle \right|,
$$

(4.20)

which can be still interpreted as an $\ell_1$-norm, in particular applied to the vector containing all the possible entries $w(i, j_k)(\mathbf{D}(j_k) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_k - i \rangle)$. The $\ell_1$-norm is know to promote sparse vectors, therefore the minimization of the function in Eq. (4.20) would try to put to zero the terms $w(i, j_k)(\mathbf{D}(j_k) - \mathbf{D}(i) - \langle \mathbf{U}(i), j_k - i \rangle)$ independently. Equivalently, the fulfillment of the constraints in Eq. (4.17) would be treated separately for each pair $(i, j)$ with $j \in \{j_1, \ldots, j_{K_i}\}$, which could potentially lead to a misfitted plane. On the other hand, the first term of our regularizer, in Eq. (4.15), can be rewritten as follows:

$$
\sum_{i} \| \mathbf{q}_i \|_2,
$$

(4.21)

which can be interpreted as an $\ell_1$-norm too, but applied to the vector $[\| \mathbf{q}_1 \|_2, \ldots, \| \mathbf{q}_i \|_2, \ldots]$. 

64
with $q_i \in \mathbb{R}^{K_i}$, and it results in a mixed norm $\ell_{1,2}$. The minimization of the function in Eq. (4.21) would try to put to zero the entry $q_i$ independently, but zeroing an entry $q_i$ translates into fulfilling all the constraints in Eq. (4.17) at once, which permits to fit a plane considering all the $K_i$ neighboring pixels.

The second term of NLTGV and our regularizer, in Eqs. (4.19) and (4.16), respectively, recall both the NLTV of Section 3.1. They both promote, directly, a piece-wise constant field $U$ and therefore promote, indirectly, a piece-wise planar inverse depth map $D$. Differently from the term of NLTGV in Eq. (4.19), our term in Eq. (4.16) aggregates the two spatial components of $U(i)$ using an $\ell_2$-norm, which leads the overall term in Eq. (4.16) to be a mixed $\ell_{1,2}$-norm as well. However, differently from the first term of our regularizer, where the $\ell_2$-norm aggregation is applied to $i$’s neighborhood, here it is limited to the spatial components of $U(i)$. The possible benefits of extending the $\ell_2$-norm aggregation at the neighborhood level is left to a future investigation.

### 4.4 Depth refinement algorithm

In this section we first present the structure of the graph underneath the regularizer in Eq. (4.15), and then the strategy adopted for the solution of the joint depth refinement and normal estimation problem presented in Eq. (4.13).

#### 4.4.1 Regularization graph construction

We assume that areas of the image $I$ sharing the same texture belong to the same object and, ideally, to the same planar surface. Based on this assumption, we build a weighted direct graph where every pixel $i$ is connected to a set of pixels $j$ which belong, potentially, to the same object. Moreover, we associate a weight to the graph edge $(i, j)$ which captures our confidence about the pixel $j$ belonging to the same object of the pixel $i$. Formally, first we define a $C \times C$ pixel search window centered at the pixel $i$, then for each pixel $j$ in the window we compute the following weight:

$$w(i, j) = \exp\left(-\frac{\|Q(i) - Q(j)\|_F^2}{2\sigma_{int}^2}\right) \exp\left(-\frac{\|i - j\|_2^2}{2\sigma_{spa}^2}\right),$$

where $Q(i) \in \mathbb{R}^{Q \times Q}$ is a square patch centered at the pixel $i$, $\|\cdot\|_F$ denotes the Frobenius norm, $\sigma_{int}, \sigma_{spa} \in \mathbb{R}_{>0}$ are tunable parameters and we recall that $i$ is a compact representation for the vector $(x, y)$. The first exponential in Eq. (4.22) provides a high weight, hence high confidence, when the patches centered at the two pixels are similar, low otherwise. The second exponential, instead, makes the weight decay as the Euclidean distance between $i$ and $j$ increases.

After the weights associated to all the pixels in the considered $C \times C$ search window have been
computed, the $K \in \mathbb{N}$ largest weights are selected and the edges between the pixel $i$ and the $K$ pixels corresponding to the selected weights are inserted in the graph. Limiting the number of connections at each pixel to $K$ reduces the computation during the minimization of the problem in Eq. (4.13), on the one hand, and it avoids weak edges that may connect pixels belong to different objects, on the other. Finally, the $K$ selected weights are normalized such that they sum to one.

4.4.2 Optimization algorithm

The problem in Eq. (4.13) is convex, but non smooth. Multiple solvers specifically tailored for these class of problems exist, such as the Backward Forward Primal Dual (FBPD) solver [48]. However, the convergence of methods like FBPD calls for the estimation of multiple parameters before the actual minimization takes place, e.g., the operator norm associated to the implicit linear operator inside the regularizer in Eq. (4.15), and this is very time consuming. The parameter estimation step cannot take place offline in our case, as the graph associated to the regularizer is scene dependent. Therefore, we decide to solve the problem in Eq. (4.13) using Gradient Descent with momentum, and in particular ADAM [58], as we empirically found it to be considerably faster (time-wise) than FBPD in our scenario.

In addition, in order to further speed up the convergence, we adopt a multi-scale approach. The noisy and possibly incomplete inverse depth map $\hat{D}$ is progressively down-sampled by a factor $r = 2$ to get $\hat{D}^\ell \in \mathbb{R}^{[H/r^\ell] \times [W/r^\ell]}$, with $\ell = 0, 1, \ldots, L - 1$, where $L \in \mathbb{N}$ is the number of scales. An instance of the problem in Eq. (4.13) is solved for each $\hat{D}^\ell$. In particular, the solution at the scale $\ell$ is up-sampled and used to initialize the solver at the scale $\ell - 1$. The multi-scale approach does not only deliver better depth maps, but it also speeds up the depth refinement algorithm, as solving the problem at scale $\ell = 0$ directly is more time demanding in our scenario.

The full algorithm is summarized in Algorithm 3. We observe that, in Algorithm 3, the up-sampling of the vector field $U^\ell$ by a factor $r$ in the multi-scale approach requires a scaling by $r^{-1}$. In fact, the up-sampling of a planar inverse depth map $d(x, y)$ with orientation $u$ leads to a new planar inverse depth $\tilde{d}(x, y)$ with orientation $r^{-1}u$. We refer to Appendix B.3 for the proof.

4.5 Experiments

In this section we test the effectiveness of our joint depth refinement and normal estimation framework. First we focus on the stereo scenario, with the train splits of the Middlebury v3 [81] and KITTI [83] stereo datasets, and address the refinement of disparity maps generated by two different stereo algorithms. Then we focus on the Multi-View Stereo (MVS) scenario, with the train split of the ETH3D [84] high-resolution dataset, and address the refinement of depth maps generated by an MVS algorithm. In the MVS scenario we make also a step further and
show the positive impact of our framework in the context of 3D reconstruction by participating to the ETH3D online benchmark.

The datasets mentioned above come with ground truth depth maps $D_{gt}$ but lack ground truth normals $N_{gt}$. Therefore, while we provide numerical results for the refined disparity and depth maps, we evaluate the estimated normal maps only visually. In particular, we plug $\nabla D_{gt}$ in Eq. (4.11) and solve the resulting system in order to get an estimate of $N_{gt}$, which can then be used for a visual comparison with the normals estimated by our framework. The gradient is computed using a $5 \times 5$ pixel Gaussian derivative kernel with standard deviation $\sigma = 0.2$ pixels. The small standard deviation permits to recover fine details, as the ground truth inverse depth map $D_{gt}$ is not affected by noise. Although this strategy does not permit a numerical evaluation, it permits to appreciate the normals estimated by our framework.
4.5.1 Middlebury and KITTI datasets

Similarly to the recent disparity refinement method in [70], we refine the disparity maps computed via Semi-Global Matching (SGM) [63] and census-based Block Matching (BM) [65]. We compare our framework to the disparity refinement method recently proposed in [70], as it also relies on a confidence map and, most importantly, it showed to outperform many other widely used disparity refinement methods, e.g., [69, 85–88], on both the Middlebury and the KITTI datasets. Moreover, since our new regularizer in Eqs. (4.15)–(4.16) resembles NLTGV [10], we compare to NLTGV as well. In particular, we replace \( g(\cdot) \) with NLTGV in our problem formulation in Eq. (4.13).

It is crucial to observe that, originally, NLTGV was introduced in the context of optical flow [10] as a general purpose regularizer, without any ambition to connect the geometry of the optical flow and the geometry of the underneath scene. In this chapter instead, we aim at modeling explicitly the joint piece-wise planarity of the inverse depth map and of the underneath scene. In fact, the mixed \( \ell_{1,2} \)-norms that we employ in both the terms of our regularizer, as opposed to the simple \( \ell_1 \)-norm of NLTGV, are carefully chosen to make our regularizer more robust in its global plane fitting.

The SGM and BM disparity maps to refine are provided by the authors in [70], who provided also their refined disparity maps and binary confidence maps. In order to carry out a fair comparison, these confidence maps are used by all the methods considered in the experiments. As described in [70], the considered binary confidence maps are the result of a learning-based framework trained on a split of the KITTI 2012 stereo dataset [89], therefore there is no bias toward the Middlebury and KITTI datasets.

Since our framework assumes a depth map at its input, we convert the disparity map to be refined into a depth map and we then convert the refined depth map back to the disparity domain, in order to carry out the numerical evaluation. The evaluation involves the **bad pixel** metric, which is the percentage of pixels with an error larger than a predefined disparity threshold, together with the **average absolute error** (avgerr) and the **root mean square error** (rms). We carry out the evaluation on all the pixels with an available ground truth, regardless of the occlusions.

Concerning the graph construction, NLTGV and our framework share the following common parameters on both the datasets: the weight parameters \( \sigma_{int} = 0.07 \) and \( \sigma_{spa} = 3 \) pixels, search window size \( C = 9 \), patch size \( Q = 3 \) and maximum number of connections per pixel \( K = 20 \). These parameters are set empirically and are shared in order to guarantee a fair comparison between the two methods, as both rely on a graph. On the other hand, the parameters \( \lambda \) and \( \alpha \) are chosen independently for each one of the two methods, and for each dataset, using a grid search; their values are specified later. Finally, we set the number of scales \( L = 2 \) and \( r = 2 \).
4.5. Experiments

**Middlebury dataset** We consider the Middlebury training dataset [81] at quarter resolution, which consists in a set of 15 indoor scenes carefully crafted to challenge modern stereo algorithms. Some scenes contain multiple untextured planar surfaces, which represent a hard challenge for stereo methods but are compliant with the model underneath our framework; other scenes are inherently non piece-wise planar instead. Due to its variety, the Middlebury dataset permits to evaluate the flexibility of our framework.

For NLTGV we set $\lambda = 7.5$ and $\alpha = 50$, regardless of the scale. For our framework and SGM disparity maps at the input, we set $\lambda = 15$ and 25 at the low and high scales, respectively. For BM disparity maps at the input instead, we set $\lambda = 10$ and 20 at the low and high scales, respectively. We set $\alpha = 3.5$ regardless of the input disparity map.

The results of our experiments on the Middlebury dataset are presented in Table 4.1 and Table 4.2 for the stereo methods SGM and BM, respectively. On average our depth refinement method outperforms the method in [70] and NLTGV [10] in four out of five error metrics (bad 0.5px, bad 2px, avgerr, rms) when the SGM stereo method is considered, and in all the metrics when BM is considered. Moreover, in the most common bad 2px metric, our framework always provides the best error regardless of the input disparity map.

In Figure 4.1 we provide the results of our experiments on the scene Piano, when the stereo methods BM is considered. The normal map associated to the input BM disparity map and to the one refined by the method in [70] are computed with the same approach adopted for the ground truth normal map, while employing $\sigma = 5$ pixels in order to handle the noise. In fact, the input BM disparity is significantly noisy, especially in the walls surrounding the piano. The method in [70] manages to decrease the error in some areas of the surrounding walls: however, since no global consistency is considered, the result is a speckled error. Instead, our method manages to approximate the surrounding walls better, using multiple planes. Finally, NLTGV fails to capture the geometry of the surrounding wall, as its relying on a simple $\ell_1$-norm makes it more sensible to outliers than our mixed $\ell_{1,2}$-norm.

The effectiveness of our framework can appreciated also in Figure 4.2, where we provide the results of our experiments on the scene Recycle when the stereo methods BM is considered. It can be observed that, differently from NLTGV, our method not only achieves a significantly better bad 2px error, but it also manages to recover the frontal part of the recycle bin despite its high error in the input disparity map.

We observe that, according to the Tables 4.1 and 4.2, our methods does not provide the best error for all the scenes. Those scenes where our refined disparity maps do not exhibit the best error can be divided roughly into two classes:

- The first class contains those scenes which are far from fulfilling our piece-wise planar assumption, therefore both NLTGV and our method are penalized inevitably: an example of this class is the scene Jadeplant, regardless of the considered stereo method. We provide the visual results of our experiments on the scene Jadeplant, when the BM
Table 4.1 – Middlebury dataset [81]: refinement of SGM [63] disparity maps. The top table reports the results for the bad pixel metric. The bottom table reports the results for the average absolute error (avgerr) and root mean square error (rms) metrics. The SGM disparity maps are refined by the method in [70], NLTGV [10] and our method. For each scene and metric, the best error is in bold.

<table>
<thead>
<tr>
<th>Scene</th>
<th>SGM [70]</th>
<th>Ours</th>
<th>SGM [70]</th>
<th>Ours</th>
<th>SGM [70]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5px</td>
<td>1px</td>
<td>2px</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ArtL</td>
<td>41.33</td>
<td>37.84</td>
<td>43.38</td>
<td>44.31</td>
<td>35.96</td>
<td>31.72</td>
</tr>
<tr>
<td>Jadeplant</td>
<td>51.95</td>
<td>50.78</td>
<td>55.64</td>
<td>55.57</td>
<td>40.15</td>
<td>37.93</td>
</tr>
<tr>
<td>Motorcycle</td>
<td>35.49</td>
<td>33.00</td>
<td>28.80</td>
<td>26.15</td>
<td>20.07</td>
<td>16.35</td>
</tr>
<tr>
<td>MotorcycleE</td>
<td>33.82</td>
<td>31.43</td>
<td>28.35</td>
<td>25.64</td>
<td>19.38</td>
<td>15.88</td>
</tr>
<tr>
<td>Piano</td>
<td>39.43</td>
<td>37.34</td>
<td>34.67</td>
<td>33.07</td>
<td>26.60</td>
<td>23.11</td>
</tr>
<tr>
<td>PianoL</td>
<td>51.14</td>
<td>49.24</td>
<td>46.30</td>
<td>44.36</td>
<td>38.34</td>
<td>35.42</td>
</tr>
<tr>
<td>Pipes</td>
<td>41.81</td>
<td>40.33</td>
<td>38.00</td>
<td>39.23</td>
<td>31.64</td>
<td>29.48</td>
</tr>
<tr>
<td>Playroom</td>
<td>52.20</td>
<td>50.03</td>
<td>50.44</td>
<td>49.83</td>
<td>37.83</td>
<td>34.50</td>
</tr>
<tr>
<td>Playtable</td>
<td>50.86</td>
<td>50.53</td>
<td>43.50</td>
<td>44.04</td>
<td>36.23</td>
<td>33.06</td>
</tr>
<tr>
<td>PlaytableP</td>
<td>38.06</td>
<td>37.02</td>
<td>30.16</td>
<td>31.67</td>
<td>26.35</td>
<td>24.04</td>
</tr>
<tr>
<td>Recycle</td>
<td>38.41</td>
<td>34.75</td>
<td>23.38</td>
<td>24.45</td>
<td>22.60</td>
<td>17.64</td>
</tr>
<tr>
<td>Shelves</td>
<td>60.81</td>
<td>58.85</td>
<td>60.03</td>
<td>57.16</td>
<td>47.63</td>
<td>44.81</td>
</tr>
<tr>
<td>Teddy</td>
<td>23.96</td>
<td>21.95</td>
<td>25.91</td>
<td>23.49</td>
<td>18.33</td>
<td>15.98</td>
</tr>
<tr>
<td>Vintage</td>
<td>54.51</td>
<td>52.74</td>
<td>43.62</td>
<td>41.67</td>
<td>38.40</td>
<td>35.59</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scene</th>
<th>SGM [70]</th>
<th>Ours</th>
<th>SGM [70]</th>
<th>Ours</th>
<th>SGM [70]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>avgerr</td>
<td></td>
<td>rms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adirondack</td>
<td>1.99</td>
<td>1.53</td>
<td>1.19</td>
<td>1.14</td>
<td>5.68</td>
<td>5.08</td>
</tr>
<tr>
<td>ArtL</td>
<td>5.13</td>
<td>3.80</td>
<td>6.00</td>
<td>4.22</td>
<td>11.07</td>
<td>8.24</td>
</tr>
<tr>
<td>Jadeplant</td>
<td>11.27</td>
<td>10.53</td>
<td>9.82</td>
<td>10.73</td>
<td>24.24</td>
<td>23.54</td>
</tr>
<tr>
<td>Motorcycle</td>
<td>2.85</td>
<td>2.32</td>
<td>1.78</td>
<td>1.92</td>
<td>8.39</td>
<td>7.59</td>
</tr>
<tr>
<td>MotorcycleE</td>
<td>2.82</td>
<td>2.25</td>
<td>1.79</td>
<td>1.89</td>
<td>8.40</td>
<td>7.46</td>
</tr>
<tr>
<td>Piano</td>
<td>2.28</td>
<td>1.48</td>
<td>1.26</td>
<td>1.02</td>
<td>5.94</td>
<td>3.41</td>
</tr>
<tr>
<td>PianoL</td>
<td>6.16</td>
<td>5.08</td>
<td>1.81</td>
<td>1.52</td>
<td>13.50</td>
<td>11.94</td>
</tr>
<tr>
<td>Pipes</td>
<td>5.85</td>
<td>5.24</td>
<td>5.20</td>
<td>5.20</td>
<td>12.64</td>
<td>11.40</td>
</tr>
<tr>
<td>Playroom</td>
<td>5.71</td>
<td>4.66</td>
<td>6.34</td>
<td>4.54</td>
<td>13.85</td>
<td>11.59</td>
</tr>
<tr>
<td>Playtable</td>
<td>3.78</td>
<td>2.88</td>
<td>2.11</td>
<td>1.66</td>
<td>9.00</td>
<td>6.97</td>
</tr>
<tr>
<td>PlaytableP</td>
<td>2.71</td>
<td>2.15</td>
<td>1.19</td>
<td>1.07</td>
<td>7.27</td>
<td>5.89</td>
</tr>
<tr>
<td>Recycle</td>
<td>1.78</td>
<td>1.35</td>
<td>1.11</td>
<td>0.95</td>
<td>4.82</td>
<td>4.11</td>
</tr>
<tr>
<td>Shelves</td>
<td>3.43</td>
<td>2.79</td>
<td>2.83</td>
<td>2.41</td>
<td>6.28</td>
<td>5.17</td>
</tr>
<tr>
<td>Teddy</td>
<td>2.69</td>
<td>1.68</td>
<td>4.23</td>
<td>1.75</td>
<td>7.68</td>
<td>5.27</td>
</tr>
<tr>
<td>Vintage</td>
<td>3.73</td>
<td>3.01</td>
<td>2.59</td>
<td>1.94</td>
<td>8.82</td>
<td>7.19</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.06</td>
<td>3.32</td>
<td>3.31</td>
<td>2.87</td>
<td>9.75</td>
<td>8.27</td>
</tr>
</tbody>
</table>

Chapter 4. Joint depth refinement and normal estimation with a planar bias
4.5. Experiments

Table 4.2 – Middlebury dataset [81]: refinement of BM [65] disparity maps. The top table reports the results for the bad pixel metric. The bottom table reports the results for the average absolute error (avgerr) and root mean square error (rms) metrics. The BM disparity maps are refined by the method in [70], NLTGV [10] and our method. For each scene and metric, the best error is in bold.

<table>
<thead>
<tr>
<th>Scene</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adirondack</td>
<td>46.97</td>
<td>34.19</td>
<td>26.70</td>
<td><strong>20.32</strong></td>
<td>34.33</td>
<td>13.71</td>
<td>15.53</td>
<td><strong>11.46</strong></td>
<td>30.11</td>
</tr>
<tr>
<td>ArtL</td>
<td>43.86</td>
<td><strong>34.49</strong></td>
<td>42.18</td>
<td>39.34</td>
<td>39.44</td>
<td><strong>29.08</strong></td>
<td>35.63</td>
<td>31.71</td>
<td>37.02</td>
</tr>
<tr>
<td>Jadeplant</td>
<td>56.74</td>
<td><strong>50.12</strong></td>
<td>56.75</td>
<td>53.09</td>
<td>47.45</td>
<td><strong>38.43</strong></td>
<td>47.35</td>
<td>44.37</td>
<td>43.59</td>
</tr>
<tr>
<td>Motorcycle</td>
<td>39.60</td>
<td>31.67</td>
<td>27.29</td>
<td><strong>24.16</strong></td>
<td>25.61</td>
<td>15.16</td>
<td>15.77</td>
<td><strong>14.82</strong></td>
<td>22.45</td>
</tr>
<tr>
<td>MotorcycleE</td>
<td>37.67</td>
<td>30.10</td>
<td>23.80</td>
<td><strong>23.80</strong></td>
<td>40.54</td>
<td>23.81</td>
<td><strong>20.33</strong></td>
<td>21.00</td>
<td>37.26</td>
</tr>
<tr>
<td>Piano</td>
<td>45.34</td>
<td>37.26</td>
<td>43.61</td>
<td><strong>35.53</strong></td>
<td>35.05</td>
<td>23.33</td>
<td>34.13</td>
<td>24.71</td>
<td>31.08</td>
</tr>
<tr>
<td>PianoL</td>
<td>60.82</td>
<td>52.92</td>
<td>51.04</td>
<td><strong>46.57</strong></td>
<td>51.78</td>
<td>40.69</td>
<td>40.32</td>
<td><strong>37.00</strong></td>
<td>48.55</td>
</tr>
<tr>
<td>Pipes</td>
<td>43.29</td>
<td>35.89</td>
<td><strong>34.91</strong></td>
<td>36.20</td>
<td>33.94</td>
<td><strong>24.46</strong></td>
<td>26.91</td>
<td>29.17</td>
<td>31.71</td>
</tr>
<tr>
<td>Playroom</td>
<td>57.53</td>
<td>50.56</td>
<td>48.12</td>
<td><strong>45.71</strong></td>
<td>46.07</td>
<td>35.60</td>
<td>35.01</td>
<td><strong>33.67</strong></td>
<td>41.20</td>
</tr>
<tr>
<td>Playtable</td>
<td>64.56</td>
<td>59.78</td>
<td>56.37</td>
<td><strong>54.00</strong></td>
<td>54.79</td>
<td>44.66</td>
<td><strong>38.24</strong></td>
<td>39.16</td>
<td>50.04</td>
</tr>
<tr>
<td>PlaytableE</td>
<td>48.51</td>
<td>36.52</td>
<td>31.67</td>
<td><strong>30.11</strong></td>
<td>40.54</td>
<td>23.81</td>
<td><strong>20.33</strong></td>
<td>21.00</td>
<td>37.26</td>
</tr>
<tr>
<td>Recycle</td>
<td>47.15</td>
<td>35.52</td>
<td>32.71</td>
<td><strong>23.90</strong></td>
<td>34.76</td>
<td>18.16</td>
<td>22.79</td>
<td><strong>11.70</strong></td>
<td>29.86</td>
</tr>
<tr>
<td>Shelves</td>
<td>64.79</td>
<td>60.93</td>
<td>59.68</td>
<td><strong>58.28</strong></td>
<td>54.25</td>
<td>48.32</td>
<td><strong>46.00</strong></td>
<td>46.63</td>
<td>47.66</td>
</tr>
<tr>
<td>Vintage</td>
<td>64.15</td>
<td>56.29</td>
<td>48.07</td>
<td><strong>42.91</strong></td>
<td>53.50</td>
<td>41.26</td>
<td>36.90</td>
<td><strong>30.66</strong></td>
<td>47.71</td>
</tr>
<tr>
<td>Avg.</td>
<td>47.48</td>
<td>39.01</td>
<td>35.69</td>
<td><strong>35.01</strong></td>
<td>37.56</td>
<td>25.83</td>
<td>26.28</td>
<td><strong>25.40</strong></td>
<td>33.98</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scene</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
<th>BM [70]</th>
<th>[10]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adirondack</td>
<td>7.07</td>
<td>1.56</td>
<td>0.99</td>
<td><strong>0.81</strong></td>
<td>13.42</td>
<td>5.28</td>
<td>2.97</td>
<td><strong>2.71</strong></td>
<td></td>
</tr>
<tr>
<td>ArtL</td>
<td>9.23</td>
<td><strong>3.91</strong></td>
<td>6.47</td>
<td>3.92</td>
<td>17.30</td>
<td><strong>8.36</strong></td>
<td>14.43</td>
<td>8.76</td>
<td></td>
</tr>
<tr>
<td>Jadeplant</td>
<td>17.65</td>
<td>9.64</td>
<td><strong>8.08</strong></td>
<td>8.38</td>
<td>35.24</td>
<td>22.57</td>
<td><strong>16.50</strong></td>
<td>18.25</td>
<td></td>
</tr>
<tr>
<td>Motorcycle</td>
<td>5.31</td>
<td>1.99</td>
<td><strong>1.50</strong></td>
<td>1.64</td>
<td>13.36</td>
<td>6.71</td>
<td><strong>4.95</strong></td>
<td>5.61</td>
<td></td>
</tr>
<tr>
<td>MotorcycleE</td>
<td>5.28</td>
<td>1.89</td>
<td><strong>1.35</strong></td>
<td>1.62</td>
<td>13.41</td>
<td>6.38</td>
<td>5.03</td>
<td>5.58</td>
<td></td>
</tr>
<tr>
<td>Piano</td>
<td>6.03</td>
<td>1.73</td>
<td>3.82</td>
<td><strong>1.02</strong></td>
<td>12.61</td>
<td>4.25</td>
<td>7.94</td>
<td><strong>2.13</strong></td>
<td></td>
</tr>
<tr>
<td>PianoL</td>
<td>12.15</td>
<td>6.73</td>
<td>2.67</td>
<td><strong>2.49</strong></td>
<td>20.11</td>
<td>13.75</td>
<td>5.04</td>
<td>5.16</td>
<td></td>
</tr>
<tr>
<td>Playroom</td>
<td>11.27</td>
<td>5.41</td>
<td>5.37</td>
<td><strong>4.50</strong></td>
<td>21.85</td>
<td>12.44</td>
<td>14.44</td>
<td><strong>11.71</strong></td>
<td></td>
</tr>
<tr>
<td>Playtable</td>
<td>9.99</td>
<td>5.13</td>
<td><strong>2.54</strong></td>
<td>2.90</td>
<td>17.54</td>
<td>10.58</td>
<td>5.71</td>
<td>6.36</td>
<td></td>
</tr>
<tr>
<td>PlaytableE</td>
<td>6.98</td>
<td>2.53</td>
<td><strong>1.23</strong></td>
<td>1.58</td>
<td>14.49</td>
<td>6.98</td>
<td>3.33</td>
<td>4.91</td>
<td></td>
</tr>
<tr>
<td>Recycle</td>
<td>6.08</td>
<td>1.55</td>
<td>1.58</td>
<td><strong>0.87</strong></td>
<td>13.53</td>
<td>4.89</td>
<td>3.84</td>
<td><strong>3.17</strong></td>
<td></td>
</tr>
<tr>
<td>Shelves</td>
<td>8.44</td>
<td>4.44</td>
<td><strong>2.51</strong></td>
<td>2.72</td>
<td>14.44</td>
<td>8.47</td>
<td>4.35</td>
<td>4.71</td>
<td></td>
</tr>
<tr>
<td>Teddy</td>
<td>4.76</td>
<td><strong>1.31</strong></td>
<td>4.93</td>
<td>3.21</td>
<td>11.45</td>
<td><strong>3.92</strong></td>
<td>12.64</td>
<td>9.26</td>
<td></td>
</tr>
<tr>
<td>Vintage</td>
<td>13.96</td>
<td>4.81</td>
<td><strong>2.53</strong></td>
<td>2.60</td>
<td>28.08</td>
<td>10.68</td>
<td>5.54</td>
<td>6.55</td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>8.41</td>
<td>3.48</td>
<td>3.35</td>
<td><strong>2.79</strong></td>
<td>17.32</td>
<td>8.58</td>
<td>7.91</td>
<td><strong>6.97</strong></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 4. Joint depth refinement and normal estimation with a planar bias

<table>
<thead>
<tr>
<th>Reference</th>
<th>BM [65]</th>
<th>[70]</th>
<th>NLTGV</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Reference Image" /></td>
<td><img src="image2.png" alt="BM Image" /></td>
<td><img src="image3.png" alt="70 Image" /></td>
<td><img src="image4.png" alt="NLTGV Image" /></td>
<td><img src="image5.png" alt="Ours Image" /></td>
</tr>
</tbody>
</table>

The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method BM [65], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.

Figure 4.1 – Middlebury [81] scene Piano. The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method BM [65], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
4.5. Experiments

Figure 4.2 – Middlebury [81] scene Recycle. The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method BM [65], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
Figure 4.3 – Middlebury [81] scene Jadeplant. The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method BM [65], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
4.5. Experiments

Figure 4.4 – Middlebury [81] scene PlaytableP. The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method SGM [63], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 2px in dark blue, error larger than 2px in yellow, missing ground truth in white. The bad 2px error percentage is reported on the bottom right corner of each disparity map.
Chapter 4. Joint depth refinement and normal estimation with a planar bias

stereo method is considered, in Figure 4.3: it can be observed that the error is concentrated on the plant branches and leaves, which are inherently non piece-wise planar, while planar surfaces are reconstructed correctly, e.g., the surface where the plant lies.

- The second class contains those scenes which fulfill the piece-wise planar assumption, but where NLTGV provides errors slightly better than our method: an example is the scene PlaytableP, when the SGM stereo method is considered. We provide the visual results of our experiments on the scene PlaytableP, when the SGM stereo method is considered, in Figure 4.4. In the figure it can be observed that our refined disparity map, despite a slightly worse error in some of the considered metrics, exhibits more meaningful normal maps than NLTGV: this is due to the mixed $\ell_{1,2}$-norm employed by our regularization in Eq.(4.16), rather than the simple $\ell_1$-norm adopted by NLTGV in Eq.(4.19).

KITTI dataset The KITTI 2015 training dataset [83] consists in a set of 200 scenes captured from the top of a moving car. As a consequence, the prevalent content of each scene are the road, possible vehicles and possible buildings at the two sides of the road. At a first glance, this content may seem to match our piece-wise planar assumption. However, in practice the buildings at the sides of the road are mainly occluded by vegetation, which is far from piece-wise planar. We select 20 scenes randomly and test our framework on them, in order to analyze its flexibility.

For NLTGV we set $\lambda = 7.5$ and $\alpha = 15$ regardless of the scale. For our framework we set $\lambda = 10$ and 20 at the lowest and highest scales, respectively, while we keep $\alpha = 15$ regardless of the scale.

The results of our experiments on the KITTI dataset are presented in Table 4.3 and Table 4.4 for the stereo methods SGM and BM, respectively. Regardless of the considered metric and stereo method, NLTGV outperforms the method in [70], while our framework outperforms all the others. Moreover, when the most common bad 3px error is considered, our framework improves the input SGM and BM disparity maps by more than 4.57% and 31.75%, respectively.

In Figure 4.5 we provide the results of our experiments on the scene 126, when the stereo method BM is considered. The method in [70], NLTGV and our framework manage all to reduce sensibly the high amount of noise that affects the input disparity map, represented by the yellow speckles. However, only NLTGV and our framework manage to preserve fine details like the pole on the left side of the image, which appears broken in the disparity map associated to [70]. Finally, our framework provides the sharpest disparity map, as NLTGV exhibits some disparity bleeding at object boundaries. This is visible on the car at the bottom right corner of the image, both by observing the disparity maps and the error masks. This is also confirmed by the numerical results, as our bad 3px error is significantly lower.
4.5. Experiments

Table 4.3 – KITTI dataset [83]: refinement of SGM [63] disparity maps. The top table reports the results for the **bad pixel** metric. The bottom table reports the results for the **average absolute error (avgerr)** and **root mean square error (rms)** metrics. The SGM disparity maps are refined by the method in [70], NLTGV [10] and our method. For each scene and metric, the best error is in bold.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>12.22</td>
<td>10.13</td>
<td><strong>9.87</strong></td>
<td>8.00</td>
<td>6.23</td>
<td><strong>4.94</strong></td>
</tr>
<tr>
<td>33</td>
<td>10.97</td>
<td>8.45</td>
<td>9.61</td>
<td>9.87</td>
<td>8.39</td>
<td>5.47</td>
</tr>
<tr>
<td>45</td>
<td>15.64</td>
<td>12.22</td>
<td>7.52</td>
<td><strong>6.79</strong></td>
<td>11.75</td>
<td>8.48</td>
</tr>
<tr>
<td>51</td>
<td>14.27</td>
<td>12.80</td>
<td><strong>9.03</strong></td>
<td>9.00</td>
<td>9.09</td>
<td>5.72</td>
</tr>
<tr>
<td>64</td>
<td>15.73</td>
<td><strong>12.66</strong></td>
<td>14.38</td>
<td>12.70</td>
<td>10.39</td>
<td>7.96</td>
</tr>
<tr>
<td>76</td>
<td>7.52</td>
<td>5.38</td>
<td>5.55</td>
<td><strong>4.50</strong></td>
<td>4.46</td>
<td>3.17</td>
</tr>
<tr>
<td>86</td>
<td>21.72</td>
<td>17.67</td>
<td>11.84</td>
<td><strong>11.19</strong></td>
<td>16.46</td>
<td>12.69</td>
</tr>
<tr>
<td>98</td>
<td>11.61</td>
<td><strong>8.18</strong></td>
<td>12.37</td>
<td>9.70</td>
<td>7.64</td>
<td>5.60</td>
</tr>
<tr>
<td>105</td>
<td>8.68</td>
<td><strong>6.25</strong></td>
<td>7.79</td>
<td>6.71</td>
<td>5.92</td>
<td>4.65</td>
</tr>
<tr>
<td>112</td>
<td>17.02</td>
<td>14.57</td>
<td>13.05</td>
<td><strong>12.86</strong></td>
<td>13.79</td>
<td>11.50</td>
</tr>
<tr>
<td>126</td>
<td>11.82</td>
<td><strong>8.75</strong></td>
<td>10.05</td>
<td>9.28</td>
<td>9.15</td>
<td>6.28</td>
</tr>
<tr>
<td>131</td>
<td>14.16</td>
<td>10.84</td>
<td>8.62</td>
<td><strong>7.41</strong></td>
<td>9.97</td>
<td>7.12</td>
</tr>
<tr>
<td>144</td>
<td>10.66</td>
<td>8.14</td>
<td>4.76</td>
<td><strong>4.24</strong></td>
<td>7.16</td>
<td>4.73</td>
</tr>
<tr>
<td>153</td>
<td>19.41</td>
<td>16.32</td>
<td>11.37</td>
<td>11.52</td>
<td>13.76</td>
<td>11.20</td>
</tr>
<tr>
<td>169</td>
<td>28.03</td>
<td>25.49</td>
<td><strong>23.23</strong></td>
<td>23.71</td>
<td>22.79</td>
<td>19.86</td>
</tr>
<tr>
<td>177</td>
<td>9.66</td>
<td><strong>8.16</strong></td>
<td>9.90</td>
<td>8.58</td>
<td>5.93</td>
<td>4.54</td>
</tr>
<tr>
<td>181</td>
<td>9.67</td>
<td>7.85</td>
<td><strong>7.82</strong></td>
<td>8.20</td>
<td>5.66</td>
<td>4.11</td>
</tr>
<tr>
<td>197</td>
<td>17.68</td>
<td>14.60</td>
<td>11.77</td>
<td><strong>11.38</strong></td>
<td>12.35</td>
<td>9.34</td>
</tr>
<tr>
<td>Avg.</td>
<td>14.25</td>
<td><strong>11.58</strong></td>
<td>10.49</td>
<td><strong>9.82</strong></td>
<td>10.11</td>
<td>7.65</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>1.68</td>
<td>1.44</td>
<td><strong>1.15</strong></td>
<td>4.93</td>
<td>4.01</td>
<td>3.71</td>
</tr>
<tr>
<td>33</td>
<td>1.56</td>
<td>1.24</td>
<td>1.24</td>
<td><strong>1.07</strong></td>
<td>4.33</td>
<td>3.00</td>
</tr>
<tr>
<td>45</td>
<td>1.82</td>
<td>1.53</td>
<td><strong>1.02</strong></td>
<td>1.02</td>
<td>4.64</td>
<td>4.11</td>
</tr>
<tr>
<td>51</td>
<td>1.96</td>
<td>1.77</td>
<td>1.24</td>
<td><strong>1.16</strong></td>
<td>5.53</td>
<td>5.02</td>
</tr>
<tr>
<td>64</td>
<td>1.69</td>
<td>1.58</td>
<td>1.59</td>
<td><strong>1.36</strong></td>
<td>4.88</td>
<td>4.00</td>
</tr>
<tr>
<td>76</td>
<td>1.23</td>
<td>0.98</td>
<td>0.95</td>
<td><strong>0.80</strong></td>
<td>3.77</td>
<td>2.62</td>
</tr>
<tr>
<td>86</td>
<td>2.59</td>
<td>2.15</td>
<td>1.52</td>
<td>6.89</td>
<td>5.92</td>
<td>5.50</td>
</tr>
<tr>
<td>98</td>
<td>2.00</td>
<td>1.51</td>
<td>1.43</td>
<td><strong>1.24</strong></td>
<td>4.98</td>
<td>3.61</td>
</tr>
<tr>
<td>105</td>
<td>1.57</td>
<td>1.01</td>
<td>1.08</td>
<td><strong>0.95</strong></td>
<td>4.37</td>
<td>2.63</td>
</tr>
<tr>
<td>112</td>
<td>2.61</td>
<td>2.48</td>
<td>1.84</td>
<td><strong>1.81</strong></td>
<td>7.22</td>
<td>7.18</td>
</tr>
<tr>
<td>126</td>
<td>2.06</td>
<td>1.72</td>
<td>1.42</td>
<td><strong>1.37</strong></td>
<td>6.21</td>
<td>5.37</td>
</tr>
<tr>
<td>131</td>
<td>1.88</td>
<td>1.52</td>
<td>1.52</td>
<td><strong>1.22</strong></td>
<td>5.24</td>
<td>4.31</td>
</tr>
<tr>
<td>144</td>
<td>1.49</td>
<td>1.21</td>
<td><strong>0.86</strong></td>
<td>0.86</td>
<td>6.30</td>
<td>4.75</td>
</tr>
<tr>
<td>153</td>
<td>2.24</td>
<td>2.00</td>
<td><strong>1.43</strong></td>
<td>1.47</td>
<td>6.37</td>
<td>5.90</td>
</tr>
<tr>
<td>169</td>
<td>3.27</td>
<td>2.83</td>
<td><strong>1.86</strong></td>
<td>1.87</td>
<td>8.06</td>
<td>7.14</td>
</tr>
<tr>
<td>177</td>
<td>1.52</td>
<td>1.31</td>
<td>1.12</td>
<td><strong>0.96</strong></td>
<td>4.60</td>
<td>3.64</td>
</tr>
<tr>
<td>181</td>
<td>1.48</td>
<td>1.29</td>
<td>0.94</td>
<td><strong>0.93</strong></td>
<td>4.17</td>
<td>3.44</td>
</tr>
<tr>
<td>197</td>
<td>2.27</td>
<td>1.83</td>
<td>1.35</td>
<td><strong>1.27</strong></td>
<td>5.82</td>
<td>4.67</td>
</tr>
<tr>
<td>Avg.</td>
<td>1.94</td>
<td>1.63</td>
<td>1.30</td>
<td><strong>1.22</strong></td>
<td>5.46</td>
<td>4.52</td>
</tr>
</tbody>
</table>

| Scene | avgerr | rms | | | | |
|-------|---------|-----| | | | |
| 28    | 1.68    | 1.15 | | | | |
| 33    | 1.56    | 1.07 | | | | |
| 45    | 1.82    | 1.02 | | | | |
| 51    | 1.96    | 1.16 | | | | |
| 64    | 1.69    | 1.36 | | | | |
| 76    | 1.23    | 0.80 | | | | |
| 86    | 2.59    | 1.52 | | | | |
| 98    | 2.00    | 1.24 | | | | |
| 105   | 1.57    | 0.95 | | | | |
| 112   | 2.61    | 1.81 | | | | |
| 126   | 2.06    | 1.37 | | | | |
| 131   | 1.88    | 1.22 | | | | |
| 144   | 1.49    | 0.86 | | | | |
| 153   | 2.24    | 1.47 | | | | |
| 169   | 3.27    | 1.87 | | | | |
| 177   | 1.52    | 1.12 | | | | |
| 181   | 1.48    | 0.94 | | | | |
| 197   | 2.27    | 1.35 | | | | |
| Avg.  | 1.94    | 1.22 | | | | |
Chapter 4. Joint depth refinement and normal estimation with a planar bias

Table 4.4 – KITTI dataset [83]: refinement of BM [65] disparity maps. The top table reports the results for the bad pixel metric. The bottom table reports the results for the average absolute error (avgerr) and root mean square error (rms) metrics. The BM disparity maps are refined by the method in [70], NLTGV [10] and our method. For each scene and metric, the best error is in bold.

<table>
<thead>
<tr>
<th>Scene</th>
<th>2px</th>
<th>Ours</th>
<th>3px</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BM</td>
<td>[70]</td>
<td>[10]</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>39.49</td>
<td>11.73</td>
<td><strong>9.86</strong></td>
<td>10.38</td>
</tr>
<tr>
<td>33</td>
<td>37.37</td>
<td>12.65</td>
<td>10.75</td>
<td><strong>10.14</strong></td>
</tr>
<tr>
<td>45</td>
<td>39.61</td>
<td>17.27</td>
<td>4.81</td>
<td><strong>4.36</strong></td>
</tr>
<tr>
<td>51</td>
<td>37.11</td>
<td>16.13</td>
<td><strong>9.19</strong></td>
<td>9.40</td>
</tr>
<tr>
<td>76</td>
<td>34.09</td>
<td>9.50</td>
<td>4.43</td>
<td><strong>3.92</strong></td>
</tr>
<tr>
<td>86</td>
<td>54.40</td>
<td>30.98</td>
<td>21.35</td>
<td><strong>20.57</strong></td>
</tr>
<tr>
<td>98</td>
<td>31.00</td>
<td>11.12</td>
<td>9.73</td>
<td><strong>8.73</strong></td>
</tr>
<tr>
<td>105</td>
<td>29.33</td>
<td>7.34</td>
<td>6.57</td>
<td><strong>5.78</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>40.96</td>
<td>16.75</td>
<td>11.09</td>
<td><strong>10.54</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>BM</th>
<th>[70]</th>
<th>[10]</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>36.26</td>
<td>7.39</td>
<td><strong>4.36</strong></td>
<td>4.60</td>
</tr>
<tr>
<td>33</td>
<td>34.02</td>
<td>8.14</td>
<td>5.73</td>
<td><strong>5.30</strong></td>
</tr>
<tr>
<td>45</td>
<td>36.86</td>
<td>13.92</td>
<td>3.40</td>
<td><strong>2.92</strong></td>
</tr>
<tr>
<td>51</td>
<td>34.29</td>
<td>11.84</td>
<td><strong>5.94</strong></td>
<td>6.08</td>
</tr>
<tr>
<td>76</td>
<td>32.18</td>
<td>6.13</td>
<td>1.88</td>
<td><strong>1.80</strong></td>
</tr>
<tr>
<td>86</td>
<td>51.81</td>
<td>27.07</td>
<td>17.63</td>
<td><strong>17.30</strong></td>
</tr>
<tr>
<td>98</td>
<td>27.58</td>
<td>6.33</td>
<td>3.66</td>
<td><strong>3.54</strong></td>
</tr>
<tr>
<td>105</td>
<td>26.99</td>
<td>5.07</td>
<td>2.98</td>
<td><strong>2.82</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>40.96</td>
<td>16.75</td>
<td>11.09</td>
<td><strong>10.54</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scene</th>
<th>avgerr</th>
<th>rms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BM</td>
<td>[70]</td>
</tr>
<tr>
<td>28</td>
<td>21.73</td>
<td>1.65</td>
</tr>
<tr>
<td>33</td>
<td>19.19</td>
<td>1.83</td>
</tr>
<tr>
<td>45</td>
<td>16.83</td>
<td>2.40</td>
</tr>
<tr>
<td>51</td>
<td>17.29</td>
<td>2.42</td>
</tr>
<tr>
<td>64</td>
<td>25.73</td>
<td>2.78</td>
</tr>
<tr>
<td>76</td>
<td>17.96</td>
<td>1.46</td>
</tr>
<tr>
<td>86</td>
<td>29.16</td>
<td>8.26</td>
</tr>
<tr>
<td>98</td>
<td>14.40</td>
<td>1.71</td>
</tr>
<tr>
<td>105</td>
<td>16.58</td>
<td>1.36</td>
</tr>
<tr>
<td>112</td>
<td>17.89</td>
<td>3.32</td>
</tr>
<tr>
<td>126</td>
<td>15.17</td>
<td>2.23</td>
</tr>
<tr>
<td>131</td>
<td>20.57</td>
<td>2.67</td>
</tr>
<tr>
<td>144</td>
<td>26.73</td>
<td>5.27</td>
</tr>
<tr>
<td>153</td>
<td>24.83</td>
<td>4.20</td>
</tr>
<tr>
<td>169</td>
<td>18.16</td>
<td>3.97</td>
</tr>
<tr>
<td>177</td>
<td>21.79</td>
<td>1.93</td>
</tr>
<tr>
<td>181</td>
<td>30.62</td>
<td>2.48</td>
</tr>
<tr>
<td>197</td>
<td>25.81</td>
<td>3.53</td>
</tr>
<tr>
<td>Avg.</td>
<td>21.12</td>
<td>2.97</td>
</tr>
</tbody>
</table>
4.5. Experiments

Figure 4.5 – KITTI [83] scene 126. The first row hosts, from left to right, the reference image and the ground truth disparity and normal maps. Each other row hosts, from left to right, the bad 2px disparity error mask and the disparity and normal map. The second row refers to the stereo method BM [65], whose disparity is refined by the methods [70], NLTGV [10], and ours, in the rows three to five, respectively. The pixels in the error maps are color coded: error within 3px in dark blue, error larger than 3px in yellow, missing ground truth in white. The bad 3px error percentage is reported on the bottom right corner of each disparity map.

4.5.2 ETH3D dataset

Large scale 3D reconstruction methods [68, 90–94] estimate the depth maps of a large number of images of the same scene, by leveraging photometric constraints among the images, and subsequently fuse them to produce a model of the scene itself. Large scale 3D reconstruction methods can largely benefit from a refinement of the estimated depth maps and can exploit the corresponding normal maps during the fusion step.

Hereafter we will consider the large scale 3D reconstruction pipeline which we proposed in [74] and which we summarize briefly here. First, a dense depth map and a dense normal map are estimated jointly, for each image of the scene, using a new MVS algorithm [74] inspired by ACMM [93]. The confidence map $M$ associated to the dense but noisy depth map $\bar{Z}$ is computed thanks to a deep neural network specifically trained to handle MVS-derived depth maps. Since depth map ranges can vary significantly in MVS applications, the network predicts the depth map confidence from the corresponding normal map and image, rather than from the depth map itself, as it typically happens in the stereo scenario. The depth map $\bar{Z}$ is then refined by our proposed framework, which produces the refined depth map $Z^*$ and the corresponding normal map $N^*$. Finally, all the refined depth maps $Z^*$ are fused together to produce the 3D reconstruction of the scene. The pipeline employs COLMAP fusion [68], which takes advantage of the estimated normals $N^*$ to reject possible outliers.
Table 4.5 – ETH3D dataset [84]: refinement of MVS-derived [74] depth maps. The first column contains the test scene. The second and third columns reports the results for the \textit{bad 2cm} and the \textit{bad 5cm} metrics, respectively. The third and fourth columns report the results for the \textit{average absolute error} (\textit{avgerr}) and the \textit{root mean square error} (\textit{rms}) metrics, respectively. For each scene and metric, the best error is in bold.

Table 4.5 compares the input MVS depth map with those refined by NLTGV and our method. The second and third columns report the percentage of pixels, computed over all the pixels of all the images in the scene, with an error within a given threshold: we consider \textit{2cm} and \textit{5cm}, which lead to the \textit{bad 2cm} and \textit{bad 5cm} metrics, respectively. On average our method outperforms NLTGV and manages to improve the input depth maps by more than 7% when the \textit{2cm} threshold, the most common in the ETH3D benchmark, is considered. In the fourth and fifth columns of the same table we provide also the \textit{average absolute error} (\textit{avgerr}) and the \textit{root mean square error} (\textit{rms}). The \textit{rms} metric is very sensitive to outliers and, especially in the \textit{Delivery Area} sequence, it highlights our improvement over the input depth map.

We provide three visual examples for each one of the three scenes considered in our experiments: \textit{Pipes} in the Figures 4.6 to 4.8, \textit{Delivery Area} in the Figures 4.9 to 4.11 and \textit{Office} in the Figures 4.12 to 4.14. In the Figures 4.6 to 4.14 our method removes the strong noise that affects the input depth maps and it estimates consistent normals. As reported in the same
4.5. Experiments

The depth maps refined by our method exhibit a significantly lower bad 5cm error compared to the input depth map. Moreover, in the examples in the Figures 4.12 to 4.14, our refined depth maps exhibit a significantly lower bad 5cm error even when compared to the depth maps refined by NLTGV.
Figure 4.6 – ETH3D [84] scene Pipes. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Figure 4.7 – ETH3D [84] scene Pipes. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Figure 4.8 – ETH3D [84] scene Pipes. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Figure 4.9 – ETH3D [84] scene Delivery Area. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Chapter 4. Joint depth refinement and normal estimation with a planar bias

Figure 4.10 – ETH3D [84] scene Delivery Area. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Figure 4.11 – ETH3D [84] scene Delivery Area. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the $5cm$ error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within $5cm$ in blue, error larger than $5cm$ in yellow, missing ground truth in white. The percentage of pixels with a depth error within $5cm$ is reported on the bottom right corner of each depth map.
Figure 4.12 – ETH3D [84] scene Office. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
4.5. Experiments

Figure 4.13 – ETH3D [84] scene Office. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
Figure 4.14 – ETH3D [84] scene Office. The first row hosts, from left to right, the reference image and the ground truth depth and normal maps. Each other row hosts, from left to right, the 5cm error map, the depth and normal maps. The second row refers to the MVS method [74], whose depth is refined by NLTGV [10] and our method in the rows three and four, respectively. The pixels in the error maps are color coded: error within 5cm in blue, error larger than 5cm in yellow, missing ground truth in white. The percentage of pixels with a depth error within 5cm is reported on the bottom right corner of each depth map.
4.5. Experiments

**3D evaluation** We now focus on our full 3D reconstruction pipeline and analyze its performance on both the ETH3D high-resolution and low-resolution datasets. We employ the same parameters adopted in the 2D evaluation but we train the confidence network using the full high-resolution train split in addition to the synthetic MVS dataset in [74]. The evaluation is carried out on the reconstructed point cloud in terms of $F_1$ score, which is computed as the harmonic average of a completeness and an accuracy term [84].

Table 4.6 – ETH3D [84] dataset: 3D reconstruction on the high and low-resolution train splits. The table reports the $F_1$ score, completeness and accuracy for the point clouds produced by our pipeline with and without our refinement step, when the 2 cm threshold is considered.

<table>
<thead>
<tr>
<th>Method</th>
<th>High-Resolution</th>
<th></th>
<th>Low-Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$F_1$ score</td>
<td>Completeness</td>
<td>Accuracy</td>
<td>$F_1$ score</td>
</tr>
<tr>
<td>ours w/o refinement</td>
<td>84.80</td>
<td>81.63</td>
<td>88.51</td>
<td>61.35</td>
</tr>
<tr>
<td>ours with refinement</td>
<td><strong>85.86</strong></td>
<td><strong>82.70</strong></td>
<td><strong>89.65</strong></td>
<td><strong>61.45</strong></td>
</tr>
</tbody>
</table>

The average $F_1$ score, completeness and accuracy associated to the train split point clouds reconstructed without and with our refinement are reported in Table 4.6, for the 2cm threshold. It can be observed that the proposed refinement achieves the best $F_1$ score, both in the high and in the low-resolution datasets. A visual example of the improvement introduced by the use of our refinement framework is provided in Figure 4.15, where the Delivery Area point clouds computed without and with our refinement are depicted. The latter is characterized by less holes in the walls and it is visibly less noisy. In addition, fine details like the red and white chain are not only preserved but denoised.

Table 4.7 reports the $F_1$ score, completeness and accuracy of the top ranked 3D reconstruction

---

Figure 4.15 – ETH3D [84] scene Delivery Area. The top row hosts the point clouds computed by our pipeline without and with our depth refinement. The bottom row hosts a detail extracted from the corresponding point clouds on the top row. It can be observed that fine structure of the red and white chain is better reconstructed in the point cloud computed by the pipeline including our refinement.
Chapter 4. Joint depth refinement and normal estimation with a planar bias

Table 4.7 – ETH3D [84] dataset: 3D reconstruction evaluation table extracted from the dataset website. The rows report the point cloud evaluation results, in terms of $F_1$ score and with $2\text{cm}$ threshold, for our proposed pipeline and for the leading published methods at the time of the thesis writing.

<table>
<thead>
<tr>
<th>Method</th>
<th>High-Resolution</th>
<th></th>
<th>Low-Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>train</td>
<td>test</td>
<td>train</td>
<td>test</td>
</tr>
<tr>
<td>Ours (DeepC-MVS)</td>
<td>85.85</td>
<td>86.80</td>
<td>61.47</td>
<td>61.99</td>
</tr>
<tr>
<td>ACMM [93]</td>
<td>78.86</td>
<td>80.78</td>
<td>55.12</td>
<td>55.01</td>
</tr>
<tr>
<td>PCF-MVS [94]</td>
<td>79.42</td>
<td>79.29</td>
<td>57.32</td>
<td>57.06</td>
</tr>
<tr>
<td>TAPA-MVS [96]</td>
<td>77.69</td>
<td>79.15</td>
<td>55.13</td>
<td>58.67</td>
</tr>
<tr>
<td>LTVRE [92]</td>
<td>61.82</td>
<td>76.25</td>
<td>53.25</td>
<td>53.32</td>
</tr>
<tr>
<td>COLMAP [68]</td>
<td>67.66</td>
<td>73.01</td>
<td>49.91</td>
<td>52.32</td>
</tr>
</tbody>
</table>

methods on the ETH3D benchmark at the time of the thesis writing, when the $2\text{cm}$ threshold is considered. It can be observed that our method, referred to as DeepC-MVS in the online table and in [74], ranks first on both the train and the test splits. Two visual examples from the high-resolution dataset test split are proposed in Figures 4.16 and 4.17, representing the scenes Statue and Old Computer, respectively. In these figures, our point cloud is compared to the ones produced by the second and third ranked 3D reconstruction methods ACMM [93] and PCF-MVS [94], respectively, and by COLMAP [68], to be considered as a baseline. In Figure 4.16, our point cloud is visibly more complete than the other point clouds, especially in the walls and in the floor areas. A similar observation holds true for Figure 4.17, where the higher completeness of our point cloud is particularly evident on the floor and on the ceiling. In addition, in both figures, our point cloud is the less noisy one.

4.6 Conclusions

We presented a framework for joint depth map refinement and normal map estimation. The proposed framework leverages the planar bias characterizing most of the human made environments, but the graph underneath our regularizer makes the framework flexible enough to handle non fully planar scenes as well. On the one hand, the proposed framework introduces a novel regularizer promoting the reconstruction of planar surfaces. On the other hand, it provides scene normals, which can potentially be useful in the context of 3D reconstruction, specifically, in the depth map fusion step [68, 74]. We compared the proposed framework with the recent method in [70], which can be considered the state-of-the-art in depth map refinement, and we showed that the proposed framework outperforms the method in [70] when the piece-wise planar assumption is met, while it leads to comparable results otherwise. We compared the proposed framework also to NLTGV [10] and we showed that our regularizer, despite sharing the same flavour of NLTGV, leads to more accurate depth maps thanks to its mixed norm which is more robust to outliers. Finally, we integrated our depth refinement and

---

5The $F_1$ scores of our pipeline for the train splits of the high and low-resolution benchmarks differ slightly from the ones in Table 4.6 as, for the participation to the benchmark, our refined depth maps have been post-processed with the sky filtering technique in [94].
4.6. Conclusions

Figure 4.16 – ETH3D [84] scene Statue. Point clouds computed by COLMAP [68] in (a), ACMM [93] in (b), PCF-MVS [94] in (c) and our pipeline including depth refinement in (d). The $F_1$ score for the 2cm threshold is reported at the bottom right corner of each point cloud.
Figure 4.17 – ETH3D [84] scene 01d. Point clouds computed by COLMAP [68] in (a), ACMM [93] in (b), PCF-MVS [94] in (c) and our pipeline including depth refinement in (d). The $F_1$ score for the 2cm threshold is reported at the bottom right corner of each point cloud.
normal estimation method in a large scale 3D reconstruction pipeline and showed its positive impact.

Finally, we observe that the refinement capabilities of the proposed framework could potentially be improved by discriminating planar and non planar surfaces ahead of the refinement. In fact, an a priori semantic segmentation of the reference image areas into the classes *planar* and *not planar* would provide a per pixel probability representing the confidence that the single pixel belongs to a planar surface or not. Therefore, the multiplier $\alpha$ in Eq. (4.15) could be set per pixel and proportional to the computed confidence, thus permitting to push the planar assumption in the planar areas and to weaken it in the non planar ones. We will consider this setup as a possible extension.
In this thesis we investigated the use of graph-based regularizers in the context of inverse problems in imaging applications. We focused on three main problems: light field super-resolution, omnidirectional stereo and depth map refinement. For each problem we proposed a novel algorithm, which leverages the modeling flexibility offered by graph-based regularizers to capture the structure of the signal to estimate.

In Chapter 2 we proposed a new light field super-resolution algorithm relying on a graph model of the light field. The presented algorithm augments the resolution of all the light field views jointly in order to preserve the light field structure: this is a crucial aspect, as the light field structure encodes the scene geometry. In particular, we showed that the use of the graph-based regularizer not only permits to preserve the light field structure in the super-resolved light field, but it also prevents the need of an a priori, accurate and expensive geometry estimation. We also showed that despite deep networks represent the state-of-the-art for single image super-resolution today, they are applied to the light field views separately: however, the light field structure must be taken into account or the super-resolved images may no longer represent a light field. A possible research direction to take advantage of deep learning techniques while avoiding to incur in the problems mentioned above could be the use of Recurrent-Neural-Networks (RNNs): these architectures could permit the sequential upsampling of the light views while preserving their consistency, as shown in [97] in the context of video super-resolution.

In Chapter 3 we improved the light field super-resolution algorithm presented in Chapter 2 and we addressed a new problem, namely omnidirectional stereo disparity estimation. Fostered by the new possible applications of omnidirectional cameras, we proposed a novel omnidirectional stereo algorithm. We showed how the geometry of an omnidirectional camera can be encoded in a graph-based regularizer to further constrain the inverse omnidirectional stereo problem. Although our algorithm represents mainly a proof of concept, replacing the pixel-wise data fidelity term with a PatchMatch-based data fidelity term [98], comparing patches rather than pixels, could significantly improve the current framework by limiting the ambiguous matchings. Moreover, still in the attempt to adopt a more discriminative data fidelity term,
the recent generalizations of the traditional 2D convolution to the sphere [99] could be used to learn a matching metric for omnidirectional cameras, as done in the perspective camera scenario [100].

In Chapter 4 we proposed a new algorithm for the joint refinement of a depth map and the estimation of the corresponding normal map. We leveraged the planar bias that characterizes most human made environments and proposed a new graph-based regularizer, which models the 3D scene in front of the camera as piece-wise planar. In particular, the graph is used to encode the probability that two pixels in the reference image correspond to the same planar surface in the scene. The overall algorithm approximates the scene by automatically fitting an arbitrary number of planes to the scene, without the need to select a set of candidate planes a priori. Since the graph-based regularizer models the planes explicitly, the proposed algorithm provides also the normals of the scene as a by product. The proposed algorithm is particularly useful in the context of Large Scale 3D Reconstruction, where the adopted MVS method is typically followed by a depth refinement step and where the depth map fusion can benefit from the normals for outlier filtering [74]. Finally, in our algorithm the level of piece-wise planarity of the scene is controlled through a simple parameter $\lambda$ which multiplies the regularizer. This strategy is not ideal, as it treats the overall scene as a whole and requires to reach a tradeoff when setting the regularizer multiplier; in fact the same scene may contain areas fulfilling accurately the piece-planar assumption and other areas that are far from planar, as in the presence of vegetation for example. The weights on the graph edges permit to enforce piece-wise planarity more where pixels are more strongly connected, hence assumed to belong to the same plane, and less otherwise. However, an a priori detection of the planar areas of the scene [101] would permit to adopt larger values of multipliers in areas classified as likely to be flat and smaller otherwise, thus avoiding the tradeoff in the selection of the regularizer multiplier. Finally, the adoption of the proposed regularizer in a PatchMatch-based global optimization formulation [98] could lead to a new MVS algorithm enforcing piece-wise planarity explicitly already at the depth estimation stage, thus avoiding the need for later refinement.

In this thesis we showed that graph-based regularizers represent a very flexible tool in inverse problems where geometry plays an important role. They can lead to state-of-the-art solutions, as shown in the case of light field super-resolution and depth refinement in Chapters 2 and 4, respectively, and can be used to handle unconventional geometrical scenarios where the images live on an arbitrary surface, as shown in Chapter 3 for omnidirectional cameras. In the era of deep learning, graph-based regularizers are an interesting example of the key role still played by hand crafted regularizers in inverse imaging problems.
A Super-Resolution Appendix

A.1 Computational Complexity

In this section we provide an estimate of the computational complexity of our super-resolution algorithm proposed in Section 2.4. This is comprised of three main steps: the construction of the graph adjacency matrix, the construction of the warping matrices, and the solution of the optimization problem in Eq. (2.15). We analyze each one of these steps separately.

In the graph construction step, the weights from each view to the eight neighboring ones are computed. Using the method in [102], the computation of all the weights from one view to the eight neighboring ones can be made independent of the size of the patch size $Q$ and computed in $O(N^2 C^2)$ operations, where $N^2$ is the number of pixels per view and $C^2$ is the maximum number of pixels in a search window. This balance takes into account also the operations required by the selection of the highest weights, which is necessary to define the graph edges. Repeating this procedure for all the $M^2$ views in the light field leads to a complexity $O(M^2 N^2 C^2)$, or equivalently $O(M^2 N^2 \alpha^2)$, as the disparity in the high resolution views grows with the super-resolution factor $\alpha$, and it is therefore reasonable to define the size $C$ of the search window as a multiple of $\alpha$.

The construction of the warping matrices relies on the previously computed weights, therefore the complexity of this step depends exclusively on the estimation of the parameter $\delta$ in Eq. (2.11). The computation of $\delta$ for each pixel in a view requires $O(N^2 C)$ operations, where $C$ is no longer squared because only 1D search windows are considered at this step. The computation of $\delta$ for all the views in the light field leads to a complexity $O(M^2 N^2 C)$, or equivalently $O(M^2 N^2 \alpha)$.

Finally, the optimization problem in Eq. (2.15) is solved via PPA, whose iterations consist in a call to the CG method (cf. steps 8-10 in Algorithm 1). Each internal iteration of the CG method is dominated by a matrix-vector multiplication with the $M^2 N^2 \times M^2 N^2$ matrix $P + (I/\beta)$. However, it is straightforward to observe that the matrix $P + (I/\beta)$ is very sparse with $O(M^2 N^2 \alpha^4)$ non-zeros entries, where we assume the size of the blurring kernel to be...
\(\alpha \times \alpha\) pixels, as in our tests in Section 2.5. It follows that the matrix-vector multiplication within each CG internal iteration requires only \(O(M^2 N^2 \alpha^4)\) operations. The complexity of the overall optimization step depends on the number of iterations of PPA, and on the number of internal iterations performed by each instance of CG. Although we do not provide an analysis of the convergence rate of our optimization algorithm, in our tests we empirically observe the following: regardless of the number of pixels in the high resolution light field, in general PPA converges after 30 iterations (each one consisting in a call to CG) while each instance of CG typically converges in only 9 iterations. Therefore, assuming the global number of iterations of CG to be independent of the light field size, we approximate the complexity of the optimization step with \(O(M^2 N^2 \alpha^4)\).

The global complexity of our super-resolution algorithm can finally be approximated with \(O(M^2 N^2 \alpha^4)\), which is linear in the number of pixels of the high resolution light field, hence it represents a reasonable complexity. Moreover, we observe that the graph and warping matrix construction steps can be highly parallelized. Although this feature would not affect the algorithm computational complexity, in practice it could lead to a significative speed up. Finally, compared to the light field super-resolution method in [29], which employs TV regularization, our algorithm turns super-resolution into a simpler (quadratic) optimization problem, and differently from the learning-based light field super-resolution method in [32] it does not require any time demanding training.
B Depth Refinement Appendix

B.1 Proof of equivalences between Eq. (4.8) and Eq. (4.10)

In Section 4.2 we already showed that Eq. (4.1) can be rewritten as Eq. (4.8), therefore it is sufficient to prove that Eq. (4.8) and Eq. (4.10) are equivalent. We start by expanding Eq. (4.10):

\[ d(x, y) = d(x_0, y_0) + (u_0^x, u_0^y), (x - x_0, y - y_0) \]
\[ = d(x_0, y_0) + u_0^x (x - x_0) + u_0^y (y - y_0) \]
\[ = (\frac{a_0}{\rho_0 f^x}) x + (\frac{b_0}{\rho_0 f^y}) y + d(x_0, y_0) - (\frac{a_0}{\rho_0 f^x}) x_0 - (\frac{b_0}{\rho_0 f^y}) y_0 \],

(B.1)

where in the last equality we used Eq. (4.9). By comparing Eq. (4.8) and Eq. (B.1) it is clear that their equality holds true if and only if the following condition is met:

\[ \left( d(x_0, y_0) -\left( \frac{a_0}{\rho_0 f^x} \right) x_0 - \left( \frac{b_0}{\rho_0 f^y} \right) y_0 \right) = \frac{1}{\rho_0} \left( c_0 - \frac{a_0 c^x}{f^x} - \frac{b_0 c^y}{f^y} \right) \].

(B.2)

We proceed by developing the right part of the equation and use the convention \( d_0 = d(x_0, y_0) \):

\[ \left( d_0 - \left( \frac{a_0}{\rho_0 f^x} \right) x_0 - \left( \frac{b_0}{\rho_0 f^y} \right) y_0 \right) = \frac{f^x f^y \rho_0 d_0 - a_0 f^y x_0 - b_0 f^x y_0}{\rho_0 f^x f^y} \]
\[ = \frac{f^x f^y \left( \frac{a_0 (x_0 - c^x)}{f^x d_0} + \frac{b_0 (y_0 - c^y)}{f^y d_0} + \frac{a_0}{d_0} \right) d_0 - a_0 f^y x_0 - b_0 f^x y_0}{\rho_0 f^x f^y} \]
\[ = \frac{a_0 f^y (x_0 - c^x) + b_0 f^x (y_0 - c^y) + f^x f^y c_0 - a_0 f^y x_0 - b_0 f^x y_0}{\rho_0 f^x f^y} \]
\[ = \frac{-a_0 f^y c^x - b_0 f^x c^y + f^x f^y c_0 - a_0 f^y x_0 - b_0 f^x y_0}{\rho_0 f^x f^y} \]
\[ = \frac{1}{\rho_0} \left( c_0 - \frac{a_0 c^x}{f^x} - \frac{b_0 c^y}{f^y} \right) \].

101
Appendix B. Depth Refinement Appendix

B.2 Close form solution of the non linear system in Eq. (4.11)

We provide the close form solution of the non linear system in Eq. (4.11), reported below in Eq. (B.3), which permits to recover the normal \( n_0 = (a_0, b_0, c_0) \) when the vector \( u(x_0, y_0) = (u^x_0, u^y_0) \) and \( d(x_0, y_0) \) are given:

\[
\begin{align*}
  u^x_0 &= (\rho_0 f^x)^{-1} a_0 \\
  u^y_0 &= (\rho_0 f^y)^{-1} b_0 \\
  a_0^2 + b_0^2 + c_0^2 &= 1.
\end{align*}
\] (B.3a, B.3b, B.3c)

For the case 1 (i.e., \( u^x_0, u^y_0 \neq 0 \)) we provide both the system solution and its derivation. For the remaining cases 2, 3 and 4 we provide only the solution, as their derivation follows the one of case 1.

B.2.1 Case 1: \( u^x_0, u^y_0 \neq 0 \)

The solution of the system in Eq. (B.3) reads as follows:

\[
\begin{align*}
  a_0 &= -|\gamma| \left( (\alpha + \beta \kappa)^2 + \gamma^2 (1 + \kappa^2) \right)^{-1} \sign(u^x_0) \\
  b_0 &= \kappa a_0 \\
  c_0 &= -\left( \alpha a_0 + \beta b_0 \right) \gamma^{-1}.
\end{align*}
\] (B.4)

The used symbols are defined as follows:

\[
\begin{align*}
  \alpha &= u^y_0 f^y (x_0 - c^x) Z_0 - f^y, \\
  \beta &= u^y_0 f^x (y_0 - c^y) Z_0, \\
  \gamma &= u^y_0 f^x f^y Z_0, \\
  \delta &= u^y_0 f^y (x_0 - c^x) Z_0, \\
  \epsilon &= u^y_0 f^x (y_0 - c^y) Z_0 - f^x, \\
  \phi &= u^y_0 f^x f^y Z_0, \\
  \kappa &= \frac{\alpha \phi - \delta \gamma}{\epsilon \gamma - \beta \phi} = (u^x_0 f^x)^{-1} u^y_0 f^y.
\end{align*}
\] (B.5)

Proof. We start by replacing \( \rho_0 \), defined in Eq. (4.6) in image coordinates, in the system of Eq. (B.3):

\[
\begin{align*}
  \begin{cases}
    (u^y_0 f^y (x_0 - c^x) Z_0 - f^y) a_0 + (u^y_0 f^x (y_0 - c^y) Z_0) b_0 + (u^y_0 f^x f^y Z_0) c_0 = 0 \\
    (u^y_0 f^y (x_0 - c^x) Z_0) a_0 + (u^y_0 f^x (y_0 - c^y) Z_0 - f^x) b_0 + (u^y_0 f^x f^y Z_0) c_0 = 0 \\
    a_0^2 + b_0^2 + c_0^2 = 1.
  \end{cases}
\end{align*}
\] (B.6)
B.2. Close form solution of the non linear system in Eq. (4.11)

Using the notation in Eq. (B.5), we can rewrite the system in Eq. (B.6) as follows:

\[
\begin{align*}
\alpha a_0 + \beta b_0 + \gamma c_0 &= 0 \quad (B.7a) \\
\delta a_0 + \epsilon b_0 + \phi c_0 &= 0 \quad (B.7b) \\
a_0^2 + b_0^2 + c_0^2 &= 1 \quad (B.7c)
\end{align*}
\]

Let us isolate \(c_0\) in Eqs. (B.7a) and (B.7b):

\[
\begin{align*}
c_0 &= -\frac{\alpha a_0 + \beta b_0}{\gamma} \quad (B.8a) \\
c_0 &= -\frac{\alpha a_0 - \epsilon b_0}{\phi} \quad (B.8b)
\end{align*}
\]

We observe that the divisions by \(\gamma\) and \(\sigma\) are legit, as \(\gamma, \sigma \neq 0\) holds true. In fact, in the definitions of \(\gamma\) and \(\sigma\) in Eqs. (B.5), the quantities \(f^x, f^y, Z_0\) are positive by definition and \(u^x, u^y \neq 0\) by assumption. Summing the Eqs. (B.8a) and (B.8b) side-wise, and isolating \(b_0\), leads to the following expression for \(b_0\):

\[
b_0 = \frac{\alpha \phi - \delta \gamma}{\epsilon \gamma - \beta \phi} a_0. \quad (B.9)
\]

We observe that \(\epsilon \gamma - \beta \phi = -u^y f^x f^y \neq 0\) holds true, hence division is legit. We replace Eq. (B.9) in Eq. (B.7c) and get the following expression for \(c_0\):

\[
c_0 = \pm \sqrt{1 - (1 + \kappa^2)} a_0^2. \quad (B.10)
\]

Now, we replace the Eqs. (B.9) and (B.10) in Eq. (B.7a) and solve for \(a_0\):

\[
\begin{align*}
a a_0 + \beta \kappa a_0 + \gamma \text{sign}(c_0) \sqrt{1 - (1 + \kappa^2)} a_0^2 \\
(\alpha + \beta \kappa) a_0 &= -\gamma \text{sign}(c_0) \sqrt{1 - (1 + \kappa^2)} a_0^2 \\
(\alpha + \beta \kappa)^2 a_0^2 &= \gamma^2 (1 - (1 + \kappa^2) a_0^2) \\
(\alpha + \beta \kappa)^2 a_0^2 + \gamma^2 (1 + \kappa^2) a_0^2 &= \gamma^2 \\
a_0^2 &= \frac{\gamma^2}{(\alpha + \beta \kappa)^2 + \gamma^2 (1 + \kappa^2)} \\
a_0 &= \pm \frac{\gamma}{\sqrt{(\alpha + \beta \kappa)^2 + \gamma^2 (1 + \kappa^2)}} \\
a_0 &= \pm \frac{|\gamma|}{\sqrt{(\alpha + \beta \kappa)^2 + \gamma^2 (1 + \kappa^2)}}
\end{align*}
\]
Appendix B. Depth Refinement Appendix

From Eq. (B.3a) we know that \( \text{sign}(a_0) = \text{sign}(\rho_0 f^x u_0^x) = -\text{sign}(u_0^x) \) holds true, as \( f^x \) is positive by definition and \( \rho_0 < 0 \) according to Eq. (4.7). We thus have the following expression for the component \( a_0 \) of the normal:

\[
a_0 = -\text{sign}(u_0^x) \frac{|\gamma|}{\sqrt{(\alpha + \beta \kappa)^2 + \gamma^2 (1 + \kappa^2)}}. \tag{B.11}
\]

Replacing Eq. (B.11) in the Eqs. (B.9) and (B.7a) provides the expressions for the components \( b_0 \) and \( c_0 \) of the normal, respectively:

\[
b_0 = \kappa a_0 \]
\[
c_0 = -\frac{\alpha a_0 - \beta b_0}{\gamma}. \]

\[\square\]

B.2.2 Case 2: \( u_0^x \neq 0, u_0^y = 0 \)

The solution of the system in Eq. (B.3) reads as follows:

\[
n_0 = \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{cases} -|\gamma| \left( \sqrt{\alpha^2 + \gamma^2} \right)^{-1} \text{sign}(u_0^x) \\ 0 \\ - (\alpha a_0) \gamma^{-1}. \end{cases} \tag{B.12}
\]

B.2.3 Case 3: \( u_0^x = 0, u_0^y \neq 0 \)

The solution of the system in Eq. (B.3) reads as follows:

\[
n_0 = \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{cases} 0 \\ -|\varphi| \left( \sqrt{\epsilon^2 + \varphi^2} \right)^{-1} \text{sign}(u_0^y) \\ - (\epsilon b_0) \varphi^{-1}. \end{cases} \tag{B.13}
\]

B.2.4 Case 4: \( u_0^x, u_0^y = 0 \)

The solution of the system in Eq. (B.3) reads as follows:

\[
n_0 = \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{cases} 0 \\ 0 \\ -1. \end{cases} \tag{B.14}
\]
B.3 Inverse depth up-sampling

In Sub-section 4.4.2 we observed that the up-sampling of the vector field \( \mathbf{U}^\ell \) by a factor \( r \) in the multi-scale approach requires a scaling by \( r^{-1} \), as happens in Algorithm 3. Formally, the up-sampling of a planar inverse depth \( d^\ell \) with slope \( u^\ell \) leads to a new planar inverse depth \( d^{\ell-1} \) with slope \( u^{\ell-1} = r^{-1} u^\ell \). Here we provide the proof.

**Proof.** Let us recall the equation of a planar inverse depth map at scale \( \ell \):

\[
d^\ell (x, y) = d^\ell (x_0, y_0) + \langle u^\ell, (x - x_0, y - y_0) \rangle,
\]

where we remove the dependencies of \( u^\ell \) from \( (x_0, y_0) \), as \( u^\ell (x_0, y_0) \) is constant for a planar inverse depth map. In the multi-scale approach, up-sampling \( d^\ell \) by a factor \( r \) is equivalent to decrease the pixel size by a factor \( r \) along both the pixel dimensions, as the camera sensor dimensions do not change. It follows that up-sampling \( d^\ell \) by a factor \( r \) is equivalent to re-sampling it with a step \( r^{-1} \):

\[
d^{\ell-1} (x, y) = d^\ell \left( \frac{x}{r}, \frac{y}{r} \right) = d^\ell (x_0, y_0) + \langle u^\ell, \left( \frac{x}{r} - x_0, \frac{y}{r} - y_0 \right) \rangle
\]
\[
= d^\ell (x_0, y_0) + \langle u^\ell, (x, y) \rangle - \langle u^\ell, (x_0, y_0) \rangle
\]
\[
= d^{\ell-1} (r x_0, r y_0) + \langle u^\ell, \left( x - r x_0, y - r y_0 \right) \rangle
\]
\[
= d^{\ell-1} (\hat{x}_0, \hat{y}_0) + \langle r^{-1} u^\ell, (x - \hat{x}_0, y - \hat{y}_0) \rangle,
\]

where in the last equality we defined \( \hat{x}_0 = r x_0 \) and \( \hat{y}_0 = r y_0 \). \( \square \)
Bibliography


Bibliography


Bibliography


**CORE STRENGTHS**


**EDUCATION**

2020  
**Signal Processing Lab LTS4 - EPFL - Switzerland**  
PhD in Electrical Engineering  
Research focused on Computer Vision problems involving multi-camera systems, using both Model-Based and Learning-Based approaches.

2013  
**University of Padua - Italy**  
M.Sc. in Computer Engineering  

**EXPERIENCE**

11/2014 – 1/2020  
**Signal Processing Lab LTS4 - EPFL - Switzerland**  
Research Assistant  
Development of algorithms for scene geometry estimation from images captured with conventional and unconventional multi-camera systems. From mathematical modeling to software optimization. Three main projects covered: super-resolution of light field data, depth refinement and normal map estimation from a single camera, omnidirectional depth estimation and view synthesis from multiple omnidirectional camera. The achievements of my research were presented at international conferences and published in the scientific literature. The omnidirectional related research was funded by Swisscom AG.  
Matlab / Python / PyTorch library / C++

8/2018 – 1/2019  
**SONY - Stuttgart Technology Center - Germany**  
Research Intern at Computational Imaging Group  
Responsible for the development of a Deep-Learning-based framework for the completion and refinement of depth maps to be integrated in the company Large Scale 3D Reconstruction pipeline. The success of the project required a close interaction and coordination with the other team members, periodic presentations to the local management and also sparse presentations to the visiting Japanese management in order to promote the project funding.  
Python / PyTorch library / C++

5/2014 – 10/2014  
**Signal Processing Lab LTS4 - EPFL - Switzerland**  
Research Intern  
Development of a multi-frame super-resolution algorithm to tackle the poor spatial resolution of portable light field camera systems in order to promote their spread in the consumer market. The developed framework and corresponding publications raised the research community attention on the importance of geometrical consistency in light field super-resolution.  
Matlab

8/2013 – 4/2014  
**Digital Image and Video Processing Lab - University of Padua - Italy**  
Research Assistant  
Development of an algorithm for the joint denoising and demosaicking of color images captured with a digital camera equipped with an arbitrary Color Filter Array. The developed framework represents one of the first works approaching the demosaicking problem using Compressed-Sensing-like techniques. In addition, the framework was tested at ARRI Inc. and proved to be competitive with their proprietary demosaicking software.  
Matlab
ADDITIONAL PROJECTS

2017
 Depth estimation for light field cameras  
EPFL - Switzerland  
Development of a multi-view-stereo algorithm for the estimation of the geometry of a scene from light field data. The problem was addressed using a Variational Bayes approach in order to estimate the scene depth map, the occlusions and the acquisition noise variance.  
Python

2017  
Super-resolution of holographic data  
EPFL - Switzerland  
Within a collaboration with the Laboratory of Applied Photonics Lab (LAPD), I worked on a software to improve the spatial resolution of a novel digital holography device characterized by unexampled compactness but inherently low resolution. The developed software allowed researchers at LAPD to increase their device resolution without trading for compactness.  
Matlab

2012  
Image compression  
University of Padua - Italy  
Development of a compression software for gray scale images. I investigated the possibility to mix Non-Uniform Vector Quantization, widely used in image compression, with Differential Pulse Code Modulation, which is typically used together with scalar quantization only. Since the non-uniform vector quantization requires a extensive learning on a large dataset of images, I carefully implemented the designed algorithm targeting computational performance.  
Matlab

2012  
Real Time System scheduler  
University of Padua - Italy  
In a team of two people we leveraged the Iterative Network Flow algorithm to develop a job scheduler for a Real Time System characterized by hard deadlines. We evaluated the scheduler on multiple simulation scenarios and developed a graphic interface to visualize the schedules computed by our software. The project involved high coordination between my team mate, responsible for the graphic interface, and me, responsible for the algorithm.  
C++ / Qt libraries

2011  
Integral Knapsack Problem on a High Performance Computer  
University of Padua - Italy  
In a team of three people we designed and implemented an algorithm to speed up the solution of the Integral Knapsack Problem using Parallel Computing. I was in charge of identifying the communications bottlenecks between the cpus and I proposed an approach to reduce the communication overhead at its minimum, thus taking fully advantage of the multiple computing units. The developed framework was tested on a machine equipped with 48 and reached a ×8 speed up over the sequential implementation.  
C / Message Passing Interface (MPI) library

2010  
Team leader in the PARIPARI project  
University of Padua - Italy  
The PARIPARI project targeted the development of a serverless and multi-purpose peer-to-peer network. It involved more than 50 programmers organized into teams, each one responsible for a specific plugin. I led the Domain Name Server and Login teams, which were responsible for the development of a fully Distributed-Hash-Table-based Domain Name Server and for the network user accounts, respectively. In particular, I designed both the plugin architectures with fault-tolerance and security as the main targets. The core functionalities of the two plugins required me to interact closely with the other teams, not only to facilitate the different plugin integration, but especially to understand the service that the other teams expected from our plugins and to fulfill them.  
Java / Extreme Programming (XP)

TECHNICAL SKILLS

Light field super-resolution / light field depth estimation / depth map refinement / multi-view-stereo large scale 3D reconstruction / omni-directional depth estimation / omni-directional view synthesis 6-degrees-of-freedom navigation / graph signal processing / deep learning for vision / data analysis sparse representations / compressed sensing / demosaicking / convex and non convex optimization

SOFTWARE SKILLS

Python (proficient) / PyTorch deep learning library (proficient) / Matlab (proficient) / Java (proficient)  
C/C++ (intermediate) / Message Passing Interface library (basic knowledge) / Adobe Illustrator (proficient)  
LaTeX (proficient) / Blender (basic knowledge)
AWARDS


LANGUAGES

**English** - Proficient in Reading, Speaking, Writing
**French** - Basic speaking abilities
**Italian** - Mother tongue

PERSONAL INFO

**Citizenship** - Italian
**Date of birth** - 28/08/1987
**Permit** - C
**Marital status** - Single

OTHER

**Driving License** - B
**Military obligations** - No

PUBLICATIONS


