Algorithms for Single-View Depth Image Estimation
by
Fangchang Ma
B.Eng., Hong Kong University of Science and Technology (2013)
S.M., Massachusetts Institute of Technology (2015)
Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Autonomous Systems
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2019
© Massachusetts Institute of Technology 2019. All rights reserved.
Algorithms for Single-View Depth Image Estimation
by
Fangchang Ma

Submitted to the Department of Aeronautics and Astronautics
on May 17, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Autonomous Systems

Abstract

Depth sensing is fundamental in autonomous navigation, localization, and mapping. However, existing depth sensors offer many shortcomings, especially low effective spatial resolutions. In order to attain enhanced resolution with existing hardware, this dissertation studies the single-view depth estimation problem – the goal is to reconstruct the dense and complete 3D structures of the scene, given only sparse depth measurements. To this end, this thesis proposes three different algorithms for depth estimation.

The first contribution is an algorithm for efficient reconstruction of 3D planar surfaces. This algorithm assumes that the 3D structure is piecewise-planar, and thus the second-order derivatives of the depth image are sparse. We develop a linear programming problem for recovery of the 3D surfaces under such assumptions, and provide conditions under which the reconstruction is exact. This method requires no learning, but still outperforms deep-learning-based methods under certain conditions.

The second contribution is a deep regression network and a self-supervised learning framework. We formulate the depth completion problem as a pixel-level regression problem and solve it by training a neural network. Additionally, to address the difficulty in gathering ground truth annotations for depth data, we develop a self-supervised framework that trains the regression network by enforcing temporal photometric consistency, using only raw RGB and sparse depth data. The supervised method achieves state-of-the-art accuracy, and the self-supervised approach attains a lower but comparable accuracy.

Our third contribution is a two-stage algorithm for a broad class of inverse problems (e.g., depth completion and image inpainting). We assume that the target image is the output of a generative neural network, and only a subset of the output pixels is observed. The goal is to reconstruct the unseen pixels based on the partial samples. Our proposed algorithm first recovers the corresponding low-dimensional input latent vector using simple gradient-descent, and then reconstructs the entire output with a single forward pass. We provide conditions under which the proposed algorithm achieves exact reconstruction, and empirically demonstrate the effectiveness of such algorithms on real data.

Thesis Supervisor: Sertac Karaman
Title: Associate Professor of Aeronautics and Astronautics
Acknowledgments

First and foremost, I am grateful to my advisor, Sertac Karaman, for his guidance. Sertac’s trust and support enabled me to explore a broad range of different research topics during my PhD program. Throughout my time as a graduate student, Sertac never ceased to impress me with his long-term vision and his commitment to fundamental research.

I have also been incredibly lucky to have both Prof. John Leonard and Prof. Nicholas Roy on my thesis committee. I am also grateful to my external evaluators, David Rosen and Michael Boulet. In particular, I would like to thank both John and David for taking the time to read through the document and providing detailed feedback that improved the clarity of the writing.

My deep appreciation goes to many brilliant collaborators that I worked with during my time at MIT - Luca Carlone, Ulas Ayaz, Guilherme Venturelli Cavalheiro, Muyuan Lin, Diana Wofk, Tien-Ju Yang, Vivienne Sze, and Jing Lin. I am particularly grateful to Luca and Ulas - the inspiring discussions with them contributed to the evolution of ideas presented in this dissertation. Sertac’s lab has also been a persistent source of fun, passion, and intellectual stimulation - David Miculescu, Ezra Tal, Igor Spasojevic, Varun Murali, Winter Guerra, Abhishek Agarwal, among many others.

I would also like to thank all my friends for the joyful memories and experiences along the way. In particular, I would like to acknowledge two of my roommates, Yili Qian and Qingkai Liang, for the invaluable help and support. It has also been an fantastic experience organizing the MIT-CHIEF events with you - Chengtao Li, Dajiang Suo, Min He, Xuanhe Fan, Chu Ma, Jingkai Chen, Yichao Pan, and so many others that I cannot even enumerate.

Finally, I would like to express my heartfelt gratitude to my parents and my girlfriend, Yumeng Yang. They have been providing me with unfailing support and continuous encouragement throughout the years, and have made many sacrifices so that I could focus on my research. This accomplishment would not have been possible without them.

The work reported in this dissertation was supported in part by the Office of Naval Research (ONR) grant N00014-17-1-2670 and the NVIDIA Corporation. Their support is gratefully acknowledged.
Contents

1 Introduction .......................................................... 19
   1.1 Overview of Depth Sensing Techniques ....................... 19
      1.1.1 Depth sensors and limitations ............................. 20
   1.2 The Single-View Depth Estimation Problem .................... 21
      1.2.1 Depth Image Representation ............................... 22
      1.2.2 Problem Formulation .................................... 22
   1.3 Application Areas ............................................. 23
      1.3.1 Depth Sensor Enhancement ................................ 23
      1.3.2 Hardware Simplification and Miniaturization ............ 23
      1.3.3 Sparse Map Densification ................................ 26
   1.4 Literature Review ............................................. 27
      1.4.1 Depth Image Inpainting and Completion .................... 27
      1.4.2 Depth Super-resolution ................................... 28
      1.4.3 Depth Prediction from Color Images ....................... 28
      1.4.4 Structure from Motion and SLAM .......................... 29
      1.4.5 Color Image Inpainting ................................... 30
   1.5 Statement of Contributions ................................... 30
   1.6 Organization .................................................. 31

2 Planar Surface Reconstruction ................................. 33
   2.1 Related Work on Compressive Sensing ......................... 34
   2.2 Preliminaries and Notation .................................. 35
   2.3 Problem Formulation .......................................... 36
      2.3.1 2D Depth Reconstruction ................................ 36
      2.3.2 3D Depth Reconstruction ................................ 38
      2.3.3 Reconciling 2D and 3D Depth Reconstruction ............... 39
   2.4 Analysis: Conditions for Exact Recovery and Error Bounds ... 40
      2.4.1 Sufficient Conditions for Exact Recovery ................ 40
      2.4.2 Depth Reconstruction from Noiseless Samples ............. 43
      2.4.3 Depth Reconstruction from Noisy Samples ................ 47
   2.5 Algorithms and Fast Solvers ................................ 51
      2.5.1 Enhanced Recovery in 2D and 3D .......................... 52
      2.5.2 Fast Solvers ............................................. 54
   2.6 Experiments .................................................. 57
      2.6.1 2D Sparse Reconstruction and Mapping ..................... 58
## List of Figures

1-1 This thesis studies the single-view depth estimation estimation. The input includes a sparse depth image and optionally a color image. The goal is to reconstruct a complete, dense depth image. ........................................ 22

1-2 Example of depth completion on lidar scans from Velodyne HDL-64. From top to bottom: RGB image, raw lidar scans, and dense depth estimation from our method described in Chapter 3. ............................. 24

1-3 Example of depth completion on Kinect [1]. Using both (a) RGB and (b) incomplete depth raw data produced by Kinect, a depth estimation algorithm can produce (c) enhanced depth image. ........................................ 24

1-4 (a) RoboBees [2] are flapping-wing micro-robots without onboard sensing capabilities. (b) Lidar-on-a-chip [3] is a new 3D Lidar with size smaller than a dime, but also produces low-resolution depth image. .......................... 25

1-5 Most state-of-the-art, real-time, visual SLAM algorithms only track and triangulate a small number of pixels in the scene. Consequently, these algorithms only maintain a sparse 3D representation of the environment. A sparse map is not ideal for robot navigation. ........................................ 26

1-6 An example of map densification using our method described in Chapter 3. From left to right: (a) RGB, (b) sparse map, (c) prediction point cloud (created by stitching RGBd predictions from each frame), and (d) ground truth. 27

2-1 We show how to reconstruct an unknown depth signal (a) from a handful of samples (b). Our reconstruction is shown in (c). Our results also apply to traditional stereo vision and enable accurate reconstruction (f) from few depth measurements (e) corresponding to the edges in the RGB image (d). Figures (a) and (d) are obtained from a ZED stereo camera. ...................... 33

2-2 (a) Example of 2D depth signal. Our goal is to reconstruct the full signal (black solid line) from sparse samples (red dots); (b) When we do not sample the corners and the neighboring points, (2.9) admits multiple minimizers, which obey the conditions of Proposition 14. .............................. 37

2-3 (a) Sampling corners and neighbors in 2D depth estimation guarantees exact recovery of the true signal. (b) Sampling only the corners does not guarantee, in general, that the solution of the $\ell_1$-minimization, namely $z^*$, coincides with the true signal $z^\diamond$. .................................................. 42

2-4 Examples of sign-consistent (shown in black) and sign-inconsistent (shown in blue) 2D depth signal. ................................................................. 44
2-5 (a) Region between a pair of twin samples, with twin samples not including corners. $d_i$ and $d_j$ represent distances between samples and the intersection of two lines. The up and down arrows represent the change of slopes. (b) Given a set of noiseless twin samples, all possible optimal solutions in 2D are contained in the envelope shown in gray. 45

2-6 (a) Illustration of a grid sample set, along with 6 non-sampled patches in white. (b) A cross section of the envelope in 3D. 46

2-7 A illustration of the active set $A$ of samples in Proposition 19. The three corners, as well as the second sample from the right, are all within the active set. A measurement $y_i$ is active if the reconstructed signal $z$ hits the boundary of $y_i$’s associated error bound. 48

2-8 (a) Illustration of the line segments (1)-(6) in Definition 22 with intersection. (b) Line segments (1)-(6) without intersection. (c) An example of 2D sign consistent $\epsilon$-envelope. 50

2-9 A toy example illustrating that while a 2D lidar produces measurements at fixed angular resolution, the resulting Cartesian coordinates are not equally spaced, i.e., $\Delta x_1 \neq \Delta x_2$. This occurs in both lidars and perspective cameras, hence motivating the introduction of the $L_1$ formulation. 54

2-10 An example of synthetic 2D signal and typical behavior of the compared techniques, naive, L1, and A1, for noise level $\varepsilon = 0.5$ m. 58

2-11 (a) Estimation errors committed by A1, L1, and naive, for increasing number of corners in the ground truth signal. A twin sample is acquired in each linear segment and measurements are noiseless. (b) Estimation errors from uniformly sampled noisy measurements, with increasing number of samples. (c) Estimation errors from uniformly sampled noisy measurements (5% of the depth data points) and increasing noise level. (d) CPU time required to solve L1 and A1 for increasing size $n$ of the 2D depth signal. 59

2-12 (a) Mapping using a conventional laser scanner, (b-c) Mapping using scans reconstructed by naive and A1 using 10 depth measurements. 60

2-13 (a) Gazebo Simulated data. (b) ZED Stereo data. (c)-(d) Kinect data. 61

2-14 Comparison between three different objective functions, L1, L1diag, and $L_1$cart, on 10 benchmarking datasets. (a) constant in $L_1$cart is chosen as $\delta = 0.1$m, (b) constant in $L_1$cart is chosen as $\delta = 0.01$m. 62

2-15 Trade-off between accuracy and speed for the NESTA solver with different parameter values $\mu_f$. As $\mu_f$ decreases, NESTA produces a more accurate solution at the cost of higher computational time. The error is computed as the average mismatch between the NESTA and cvx solutions. 64

2-16 Comparison between NESTA ($\mu_f = 0.001$) and cvx. Estimation errors and timing are shown for (a-b) increasing number of samples, (c-d) increasing measurement noise, (e-f) increasing size of the $N \times N$ depth signals. NESTA achieves comparable reconstruction errors, while offering a significant speedup. 65
2-17 The first row is an example of sparse depth reconstruction on Gazebo simulated data: (a) RGB image, (b) uniformly drawn sparse samples, and (c) reconstruction using \( L_{\text{ldiag}} \). The second row is an example on Kinect \( K1 \) data: (d) RGB image, (e) sparse samples on a grid, and (f) reconstruction using \( L_{\text{ldiag}} \). 66

2-18 Qualitative results on Gazebo dataset: 3 examples of reconstructed depth signals using the proposed approaches (L1, \( L_{\text{ldiag}} \)) and a naive linear interpolation (naive). For each example we show the reconstruction from 2% uniformly drawn depth measurements. We also show the reconstruction for the case in which we can access the depth corresponding to (appearance) edges in the RGB images. 67

2-19 Qualitative results on ZED: 3 examples of reconstructed depth signals using the proposed approaches (L1, \( L_{\text{ldiag}} \)) and a naive linear interpolation (naive). For each example we show the reconstruction from 2% uniformly drawn depth measurements. We also show the reconstruction for the case in which we can access the depth corresponding to (appearance) edges in the RGB images. 68

2-20 Reconstruction errors for increasing percentage of uniform samples, and for different datasets. (a) and (b) are reconstructions on the Gazebo dataset, using noiseless and noisy (noise bounded by \( \varepsilon = 0.1 \text{m} \)) samples, respectively. (c) reconstruction on the ZED dataset, (d) reconstruction on the K1 dataset. (e) comparison on all datasets. 70

2-21 Examples of reconstruction from 5% uniformly random samples on the Middlebury disparity dataset, using 4 different algorithms: naive, \( L_{\text{ldiag}} \), CSR [4], and WT+CT [5]. The proposed algorithm, \( L_{\text{ldiag}} \), is able to preserve sharper boundaries and finer details, while not creating jagged edges as in naive. 71

2-22 Reconstructed depth profile of the “Art” image from the Middlebury dataset from 5% uniform random measurements, visualized as point cloud. Red indicates errors and green highlights good reconstruction quality. In comparison against naive and WT+CT [5], \( L_{\text{ldiag}} \) produces the most complete structures with the least outliers. 72

2-23 Super-resolution depth imaging. The up-scale factor is 24.79 in this example. 75

3-1 CNN architecture for NYU-Depth-v2 and KITTI datasets, respectively. The network for KITTI has fewer layers and channels, since KITTI has a larger image size than NYU-Depth-v2 and thus demands higher memory consumption. Cubes are feature maps, with dimensions represented as \#features@height×width. The encoding layers in blue consist of a ResNet [6] and a 3×3 convolution. The decoding layers in yellow are composed of 4 upsampling layers (\text{UpProj}) followed by a bilinear upsampling. 78

3-2 Predictions on NYU-Depth-v2. From top to bottom: (a) RGB images; (b) RGB-based prediction; (c) d prediction with 200 and no RGB; (d) \( \text{RGBd} \) prediction with 200 sparse depth and RGB; (e) ground truth depth. 83
3-3 Impact of number of depth sample on the prediction accuracy on the NYU-Depth-v2 dataset. Left column: lower is better; right column: higher is better.

3-4 Impact of number of depth sample on the prediction accuracy on the KITTI dataset. Left column: lower is better; right column: higher is better.

3-5 Application in sparse SLAM and visual inertial odometry (VIO) to create dense point clouds from sparse landmarks. (a) RGB (b) sparse landmarks (c) ground truth point cloud (d) prediction point cloud, created by stitching RGBd predictions from each frame.

3-6 An illustration of the self-supervised training framework, which requires only a sequence of color images and sparse depth images. White rectangles are variables, red is the depth network to be trained, blue are deterministic computational blocks (without learnable parameters), and green are loss functions.

3-7 Comparison between different training methods (best viewed in color). The photometric loss provides supervision at the top, where the semi-dense annotation does not contain labels.

3-8 Prediction error against number of input depth samples, for both spatial patterns (uniform random sub-sampling and Lidar scan lines). (a) When trained with semi-dense ground truth, the depth completion error decreases as a power function $c x^p$ of the number of input depth measurements, for some $c > 0, p < 0$. (b) The self-supervised framework is effective with sufficiently many measurements (at least 4 scanlines, or the equivalent number of samples to 2 scanlines when input is uniformly sampled).

4-1 Recovery of the input latent code $z$ from under-sampled measurements $y = AG(z)$ where $A$ is a sub-sampling matrix and $G$ is an expanding generative neural network. We prove that $z$ can be recovered with guarantees using simple gradient-descent methods under mild technical assumptions. Best viewed in color.

4-2 Illustration of a single transposed convolution operation. $f_{i,j}$ stands for $i^{th}$ filter kernel for the $j^{th}$ input channel. $z$ and $x$ denote the input and output signals, respectively. (a) The standard transposed convolution represented as linear multiplication. (b) With proper row and column permutations, the permuted weight matrix has a repeating block structure.

4-3 Distribution of the kernel weights from every layer in a trained convolutional generative network. The trained weights roughly follow a zero-mean gaussian distribution.

4-4 The landscape of the cost function $J(z)$ for deconvolutional networks with (a) ReLU, (b) Sigmoid, and (c) Tanh as activation functions, respectively. There exists a unique global minimum.

4-5 We demonstrate recovery of latent codes on a generative network trained on the MNIST dataset. From top to bottom: ground truth output images; partial measurements with different sampling masks; reconstructed image using the recovered latent codes from partial measurements. The recovery of latent codes is exact using simple gradient descent.
4-6 recovery of latent codes on a generative network trained on the CelebA dataset. From top to bottom: ground truth output images; partial measurements with different sampling masks; reconstructed image using the recovered latent codes from partial measurements. The recovery of latent codes is exact using simple gradient descent. ............................................ 106

A-1 Region between a pair of twin samples, with a twin sample including a corner. 119
A-2 Example of matrix $(\Delta^T)_{\mathcal{M}}$ and corresponding column partition $(\Delta^T)_{\mathcal{M},V}$, $(\Delta^T)_{\mathcal{M},H}$. .......................................................... 120
A-3 Illustration of the upper bound $\bar{z}$ of 2D sign consistent $\epsilon$-envelope for all possible orientations of the line segments (1), (3) and (5) defined in Definition 22. .......................................................... 125
B-1 Plot of function $g(\cdot)$ defined in Section B.2 .............................. 133
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Summary of the key theoretical results.</td>
<td>41</td>
</tr>
<tr>
<td>2.2</td>
<td>Reconstruction accuracy and computational time comparing naive, L1diag, CSR [4], and WT+CT [5].</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>L1diag consistently outperforms all other methods in accuracy, and performs robustly even with aggressively low number of measurements.</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>Our proposed solution outperforms state-of-the-art deep-learning based method [7] on the NYU test dataset. The input depth is sampled uniform randomly (1%) from the ground truth.</td>
<td>74</td>
</tr>
<tr>
<td>3.1</td>
<td>Evaluation of loss functions, upsampling layers and the first convolution layer. RGBd has an average sparse depth input of 100 samples. (a) comparison of loss functions is listed in Row 1 - 3; (b) comparison of upsampling layers is in Row 2 - 4; (c) comparison of the first convolution layers is in the 3 bottom rows.</td>
<td>80</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparison with state-of-the-art on the NYU-Depth-v2 dataset. The values are those originally reported by the authors in their respective paper.</td>
<td>82</td>
</tr>
<tr>
<td>3.3</td>
<td>Comparison with state-of-the-art on the KITTI dataset. The Make3D values are those reported in [8].</td>
<td>82</td>
</tr>
<tr>
<td>3.4</td>
<td>Evaluation of the self-supervised framework on the validation set.</td>
<td>91</td>
</tr>
<tr>
<td>B.1</td>
<td>Summary of notation</td>
<td>129</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The recovery of scene geometry and 3D structure has long been one of the core problems in computer vision, robotics, and mapping. A significant amount of past efforts (namely multi-view geometry, Structure from Motion, and Simultaneous Localization and Mapping) has been devoted on 3D reconstruction from multiple viewing angles. Orthogonally, other work (e.g., single-view metrology) has been focusing on estimating geometry from one single frame of sensor data. This dissertation fits into this line of research and studies the problem of single-view depth image estimation. Our goal is to estimate the 3D structure of a scene, represented as a depth image, based on incomplete depth samples (for instance, raw measurements from a low-resolution lidar sensor) captured at a single viewing angle. Specifically, we address the following question: is it possible to reconstruct a dense, complete depth image from sparse and incomplete depth samples?

In general, the answer is negative since depth reconstruction from incomplete measurements is an ill-posed inverse problem, meaning there exists an infinite number of possible solutions. However, in this thesis, by posing proper modeling assumptions regarding the scene geometry, we are able to develop efficient computational algorithms for reconstruction of the 3D structure. In particular, we propose 3 different algorithms for single-view depth estimation, each based on a different set of modeling assumptions. We empirically demonstrate their performance on real data, and provide conditions for exact reconstruction under rigorous proofs.

1.1 Overview of Depth Sensing Techniques

Depth sensing, which refers to measuring distances from a sensor to the surrounding environments, is fundamental in robotics and computer vision. It is an essential component in many engineering applications, including mapping [9, 10, 11, 12, 13], localization [14, 15, 16], obstacle avoidance [17, 18], and etc. The measured depth also serves as part of the input to a variety of higher-level computer vision tasks, including object recognition [19, 20, 21], detection [22, 23, 24, 25, 26], segmentation [27, 28, 29, 30], and beyond.

Historically, depth sensing techniques have been a key driver in the development of robotics. For instance, in 1970s, Moravec [31] developed the first stereo vision system for
the Stanford Cart, which became the first mobile robot to navigate around simple polygonal objects over 20 meters. Over the period of 1980s to early 2000s, sonars and 2D lidars inspired a large number of pioneering work on robot navigation [32, 33] and SLAM (Simultaneous Localization and Mapping) [34, 35, 36]. In 2007, Velodyne Acoustics released the HDL-64E [37], the first 3D lidar. It was mounted atop five of the six autonomous vehicles that competed in the 2007 DARPA Urban Challenge [38] as an essential component in their perception systems [39, 40, 41, 42]. 3D lidars remain crucial to most autonomous driving cars today, as well as in the foreseeable future. In 2010, Microsoft introduced the structured-light based depth sensor Kinect. Kinect provides high-frequency, accurate and dense depth measurements, enabling high-speed MAV (Micro Aerial Vehicle) flights in cluttered, GPS-denied environment [13].

1.1.1 Depth sensors and limitations

Mainstream depth sensing techniques include sonar (Sound Navigation and Ranging) [43], radar (Radio Detection and Ranging) [44, 45, 46], lidar (Light Detection and Ranging) [47, 48], as well as stereo vision [49, 50] and its variants (e.g., structured-light sensors such as Microsoft Kinect V1 [51]). However, these depth sensing techniques offer many shortcomings. No existing depth sensors can attain both high RRA (Range, Resolution and Accuracy) and low SWaP (Size, Weight and Power). Such limitations render existing depth sensors unfit for a large number of hardware platforms, including miniaturized robots and embedded systems.

**Sonar** Sonars were motivated by the impressive echolocation capabilities of bats in using sound for aerial navigation [52]. Specifically, an active sonar creates a pulse of sound and then measures the time difference between transmission and reception of the reflected pulse. The distance to an object can then be derived as the product of one-half of the round-trip time with the speed of sound. This method for measuring distances is called ToF (time-of-flight). A typically sonar produces only planar measurements [43]. Sonars suffer from poor directionality, frequent misreadings, and data corruption due to specular reflections. The use of a sonar range finder represents a worst-case scenario for robot localization [53].

**Radar** Radars were initially developed for military purposes during World War II. Later radars find highly diverse applications, including automotive collision avoidance [54] and power line detection [55]. Based on the same ToF strategy as sonars, radars obtain distance measurements by first emitting a signal (radio waves) and then measuring the round-trip time. In addition, if an object is moving towards or away from the radar, its speed can be inferred from the change of received radio wave frequency caused by the Doppler effect. Furthermore, radars are robust against environmental influences including extreme weather and change of temperatures and lighting conditions [56]. However, in order to ensure interference suppression, only a few antennas are used on a radar sensor and mechanical scanning is used to sweep across a fixed angular range [56]. The output radar image has a low resolution.

**Lidar** Lidars are also ToF sensors. They illuminate a target with pulsed laser light, and
the measure the round-trip time of the reflected pulses. Most lidars on vehicles today obtain a wide field of view by rotating laser beams. 3D lidars available today are bulky, heavy, power-consuming, and cost-prohibitive\(^1\). In addition, lidar has its limitations in adverse weather conditions, such as snow, rain, and fog [57]. Furthermore, they have a fixed angular resolution, and thus provide only sparse measurements for distant objects, as illustrated in Figure 1-2.

In order to address the current limitations, extensive efforts have been invested into the development of new lidar hardware. For instance, companies in the lidar space are competing to produce compact, narrow field-of-view solid-state lidars\(^2\) [59] as a complement to the existing high-end, 360-degree lidar. Research and development work to create tiny lidars is also underway in academia. For example, Poulton et al. [3] are developing a lidar-on-a-chip system with a size smaller than a dime.

**Stereo Vision** Stereo vision extracts 3D information by comparing a pair of slightly different views of the same scene, similar to the way human eyes work. Typically, two horizontally displaced cameras are used for triangulation. A disparity image between these two cameras can be computed using either local comparison approaches [19, 60] or global methods [61, 62]. However, stereo vision is computationally demanding. It also requires a large baseline and sufficiently dense textures for accurate disparity inference. Furthermore, stereo vision faces the same difficulty with precipitation and obscurants as lidars [57].

**Structured-light Sensors** Structured-light-based depth sensors (e.g. Kinect) are variants of the traditional stereo cameras. Take Microsoft Kinect V1 for an example: instead of using two cameras, Kinect V1 uses an IR (infrared) projector and an IR camera for depth sensing. The IR projector emits a special dot pattern, which brings robustness in areas with poor visual texture [51]. However, Structured-light-based depth sensors can be sensitive to sunlights, and are range-limited by their projectors (typically no more than 10 meters).

### 1.2 The Single-View Depth Estimation Problem

The single-view depth estimation problem is illustrated in Figure 1-1. In this problem, the input to the system includes a sparse depth image and optionally a RGB image of the same scene. A sparse depth image is a depth image where only a small set of pixels are measured, whilst the other pixels are simply encoded as zeros (corresponding to no sensor measurements). Sparse depth images are typically found in lidar measurements, or 3D landmarks tracked in a SLAM algorithm. The goal of is to predict a complete and dense depth image of the scene, given only the incomplete sensor measurements and information.

In particular, the problem is often dubbed *depth completion* or *depth inpainting* when the input includes sparse depth (with or without color). In comparison, when only a color image is available, the problem is often referred to as *depth prediction*. In this thesis, we focus on the depth completion problem, since in robotic applications it is relatively easy to

\(^1\)The top-of-the-range 3D LiDARs cost up to $75,000 per unit.

\(^2\)Solid-state lidars are single-chip lidars with no moving parts. They rely on optical phased arrays for beam steering [58].
Figure 1-1: This thesis studies the single-view depth estimation estimation. The input includes a sparse depth image and optionally a color image. The goal is to reconstruct a complete, dense depth image.

obtain sparse depth samples, either from a low-resolution depth sensors or from triangulations due to motions.

1.2.1 Depth Image Representation

A depth image encodes the geometry information of the scene in the format of an image. A depth image is formed by projecting a 3D world point \( P = (X, Y, Z) \) into the image plane using a perspective transformation. The value of each depth pixel is simply \( Z \) (i.e., the distance of \( P \) along the optical axis). This image formation process can be modeled as a typical pinhole camera. Some alternative encoding of geometry include inverse depth representation (i.e., each pixel stores \( 1/Z \) rather than \( Z \)), or disparity between a pair of stereo cameras. Since there is no clear advantage of one representation over another, we adopt the linear representation \( Z \) throughout this thesis for consistency.

1.2.2 Problem Formulation

Let \( Z \in (0, \infty)^{h \times w} \) denote the target ground truth signal (a 2D depth image of height \( h \) and width \( w \)), which we do not have direct access to. Instead, we only observe a sparse depth image \( Y \), given some sampling pattern known \textit{a priori}. Consequently, the depth completion problem can be formally stated as follows.

Problem 1 (Matrix form). \textit{Find the target depth signal} \( Z \), \textit{given both the binary sampling mask} \( B \in \{0, 1\}^{h \times w} \) \textit{and the sparse depth measurements}

\[
Y = B \circ (Z + E),
\]
where \( E \in \mathbb{R}^{h \times w} \) represents unknown measurement noise (unless stated otherwise), and \( \circ \) is the Hadamard product (i.e., element-wise product) between two matrices.

Alternatively, we can rephrase the same problem in matrix-vector multiplication form. Let \( y \in (0, \infty)^m \) be a vector that includes all the \( m \) samples (i.e., all zeros in \( Y \) have been excluded), then Problem 1 can be rewritten as

**Problem 2** (Matrix-vector form). Find the target depth signal \( Z \), given both the sub-sampling matrix \( A \in \{0, 1\}^{m \times (h \times w)} \) and the sparse depth measurements

\[
y = A \cdot \text{vec}(Z) + \eta,
\]

where \( \eta \in \mathbb{R}^m \) represents unknown measurement noise (unless stated otherwise), and \( \text{vec}() \) is the vectorization operation.

Note that Problem 1 and Problem 2 are equivalent statements of the same problem this dissertation studies. They both represent a mathematical abstraction of the depth reconstruction problem, which can be application- and sensor-dependent in practice.

### 1.3 Application Areas

In this section we give a brief overview of some use cases of single-view depth estimation. In particular, we will look at depth sensor enhancement, hardware simplification and miniaturization, as well as sparse map densification.

#### 1.3.1 Depth Sensor Enhancement

Existing depth sensors are considered to be low-resolution and noisy, as is discussed in Section 1.1.1. The single-view depth estimation problem this dissertation studies can enhance the quality of depth images captured by such depth sensors. For instance, lidar (Light Detection and Ranging) produces only sparse measurements, see Figure 1-2. Similarly, Kinect generates depth maps with many missing pixels, see Figure 1-3(b), due to occlusions and other factors.

The incomplete raw depth data can lead to computational challenges and lower accuracy for high-level algorithms in the data processing pipeline. Therefore, depth completion and inpainting is usually required as a pre-processing step.

#### 1.3.2 Hardware Simplification and Miniaturization

Another motivation for this thesis project is to enable onboard sensing and navigation capabilities for miniaturized robotic systems. Recent years have witnessed a growing interest towards miniaturized robots, for instance the RoboBee [63], Piccolissimo [64], the DelFly [65, 66], the Black Hornet Nano [67], Salto [68]. These robots are usually palm-sized (or even smaller), can be deployed in large volumes, and provide a new perspective on societally relevant applications, including artificial pollination, environmental monitoring,
Figure 1-2: Example of depth completion on lidar scans from Velodyne HDL-64. From top to bottom: RGB image, raw lidar scans, and dense depth estimation from our method described in Chapter 3.

Figure 1-3: Example of depth completion on Kinect [1]. Using both (a) RGB and (b) incomplete depth raw data produced by Kinect, a depth estimation algorithm can produce (c) enhanced depth image.
and disaster response. Despite the rapid development and recent success in control, actuation, and manufacturing of miniature robots, on-board sensing and perception capabilities for such robots remain a relatively unexplored, challenging open problem. These small platforms have extremely limited payload, power, and on-board computational resources, thus preventing the use of standard sensing and computation paradigms.

(a) RoboBees
(b) lidar on a chip

Figure 1-4: (a) RoboBees [2] are flapping-wing micro-robots without onboard sensing capabilities. (b) Lidar-on-a-chip [3] is a new 3D lidar with size smaller than a dime, but also produces low-resolution depth image.

In the last two decades, a large body of robotics research focused on the development of techniques to perform inference from data produced by “information-rich” sensors (e.g., high-resolution cameras, 2D and 3D laser scanners). A variety of approaches has been proposed to perform geometric reconstruction using these sensors, for instance see [69, 70, 71] and the references therein. On the other extreme of the sensor spectrum, applications and theories have been developed to cope with the case of minimalistic sensing [72, 73, 74, 75]. In this latter case, the sensor data is usually not metric (i.e., the sensor cannot measure distances or angles) but instead binary in nature (e.g., binary detection of landmarks), and the goal is to infer only the topology of the (usually planar) environment rather than its geometry. This work studies a relatively unexplored region between these two extremes of the sensor spectrum.

This thesis complements recent work on hardware and sensor design, including the development of lightweight, small-sized depth sensors. For instance, a number of ultra-tiny laser range sensors are being developed as research prototypes (e.g., the dime-sized, 20-gram laser of [76], and an even smaller lidar-on-a-chip system with no moving parts [77]), while some other distance sensors have already been released to the market (e.g., the TerraRanger’s single-beam, 8-gram distance sensor [78], and the LeddarVu’s 8-beam, 100-gram laser scanner [79]). These sensors provide potential hardware solutions for sensing on micro (or even nano) robots. Although these sensors meet the requirements of payload and power consumption of miniature robots, they only provide very sparse and incomplete depth data, in the sense that the raw depth measurements are extremely low-resolution (or even provide only a few beams). In other words, the output of these sensors cannot be utilized directly in high-level tasks (e.g., object recognition and mapping), and the need to
reconstruct a complete depth profile from such sparse data arises.

### 1.3.3 Sparse Map Densification

In many robotic navigation problems, Simultaneous Localization and Mapping (SLAM) is a critical task of the system. SLAM provides not only the position of the robot, but also a map of the surrounding environments. Most state-of-the-art, real-time, visual SLAM algorithms (e.g., ORB-SLAM [80], LSD-SLAM [81], PTAM [82, 83]) only track and triangulate a small number of pixels in the scene. Consequently, these algorithms only maintain a sparse 3D representation of the environment, see Figure 1-5. However, such sparse maps are far from ideal, because it provides only partial information regarding the scene geometry.

![Figure 1-5: Most state-of-the-art, real-time, visual SLAM algorithms only track and triangulate a small number of pixels in the scene. Consequently, these algorithms only maintain a sparse 3D representation of the environment. A sparse map is not ideal for robot navigation.](image)

As a remedy, depth estimation techniques can be applied as a downstream component of the SLAM algorithm. Specifically, as post-process step, depth estimation algorithms create a dense representation of the environment from the sparse map tracked by SLAM. This solution creates a map of significantly higher density, without incurring any extra computational overhead in the SLAM algorithm. An example is demonstrated in Figure 1-6. The estimated dense representation is not only visually more comprehensible, but also improves the safety and efficiency of robot navigation missions.
1.4 Literature Review

This thesis intersects several lines of research across fields. In particular, the most relevant topics include depth image inpainting, super-resolution and prediction. RGB image inpainting also shares some similarities to the depth reconstruction problem. In addition, this dissertation also relates to 3D reconstruction problems such as structure from motion.

1.4.1 Depth Image Inpainting and Completion

Depth completion [84, 85] and depth inpainting [86] refer to the problem of filling in missing pixels in a depth image. Both two terms are used interchangeably in some context. The depth completion problem is typically sensor- and input-dependent. Consequently, the problem faces vastly different levels of algorithmic difficulty, under different input modalities (with color images for guidance [87, 88] vs. without [89]) and depth densities (dense depth input [85, 87, 90] vs. sparse depth measurements [7, 91]).

High-density Input Depth completion for structured light sensor (e.g., Microsoft Kinect) [1] is more often referred to as depth inpainting [86], or depth enhancement [85, 87, 90] when noise is taken into account. The task is to fill in small missing holes in the relatively dense depth images. This problem is relatively easy, since a large portion of pixels (typically over 80%) are observed. Consequently, even simple filtering-based methods [85] can provide good results.

Sparse Input In comparison, the completion problem becomes significantly more challenging with a low depth density (e.g., below 5%), which is the scenario that we would like to study in this thesis. For instance, one application is to create a dense disparity map from sparse triangulated feature correspondences in stereo vision [4, 5]. [4] exploit the sparsity of the disparity maps in the Wavelet domain. The dense reconstruction problem is then posed as an optimization problem that simultaneously seeks a sparse coefficient vector in the Wavelet domain while preserving image smoothness. They also introduce a conjugate subgradient method for the resulting large-scale optimization problem. Liu et al. [5] empirically show that a combined dictionary of wavelets and contourlets produces a better sparse
representation of disparity maps, leading to more accurate reconstruction.

Another application is lidar depth completion. For instance, in the KITTI dataset [89], the projected depth measurements from a Velodyne lidar onto the camera image space account for only roughly 4% of all pixels. Depth completion using these sparse lidar measurements has attracted a significant amount of recent interest. Ku et al. [92] developed a simple and fast interpolation-based algorithm that runs on CPUs. Uhrig et al. [89] proposed sparse convolution, a variant of regular convolution operations with input normalizations, to address data sparsity in neural networks. Eldesokey et al. [93] improved the normalized convolution for confidence propagation. Chodosh et al. [94] incorporated the traditional dictionary learning with deep learning into a single framework for depth completion. Cadena et al. [95] developed a multi-modal auto-encoder to learn from three input modalities, including RGB, depth, and semantic labels. In their experiments, Cadena et al. [95] used sparse depth on extracted FAST corner features as part of the input to the system to produce a low-resolution depth prediction. The accuracy was comparable to using RGB alone. Liao et al. [96] studied the use of a 2D laser scanner mounted on a mobile ground robot to provide an additional reference depth signal as input and obtained higher accuracy than using RGB images alone.

1.4.2 Depth Super-resolution

Both the depth completion and inpainting problems also find close connection to depth super-resolution [97, 98, 99, 100, 101, 102]. Depth super-resolution is the problem of creating a higher resolution depth image using only a low-resolution depth image as input, and can be treated as a special case of depth completion and inpainting where the sparse depth input has a grid-structure. Hui et al. [103] propose a Mult-Scale Guided convolution network (MSG-Net), which upsamples the high-frequency components of a low-resolution depth image progressively. The high-resolution depth image is reconstructed by merging the upsampled high-frequency component with the low-frequency counterpart. Uhrig et al. [89] propose a new convolution operation which normalizes the output with the number of non-zero input pixels, and show experimentally that this operation exhibit high level of sparsity-invariance.

Limitations Prior work in depth completion mostly focuses on the high-density regime. However, the problem becomes significantly more challenging with a low-density, sparse input. Depth completion with sparse input has traditionally received relatively little attention due to its complexity, but has started to attract attentions since very recently. Most of the existing efforts are on an ad-hoc basis, and lack performance guarantees.

1.4.3 Depth Prediction from Color Images

Color-based depth image prediction, which refers to the inference of 3D depth information from a single 2D color image, is an active research area in computer vision and robotics.

Early Work Early works on depth estimation using RGB images usually relied on hand-crafted features and probabilistic graphical models. For instance, Saxena et al. [104] esti-
mated the absolute scales of different image patches and inferred the depth image using a Markov Random Field model. Non-parametric approaches [105, 106, 107, 108] were also exploited to estimate the depth of a query image by combining the depths of images with similar photometric content retrieved from a database.

**Deep-Learning Approaches**  The state-of-the-art RGB-based depth prediction methods exclusively use deep-learning based methods to train a convolution neural network using large-scale datasets [109, 110, 111]. Eigen et al. [8] suggest a two-stack convolutional neural network (CNN), with one predicting the global coarse scale and the other refining local details. Eigen and Fergus [110] further incorporate other auxiliary prediction tasks into the same architecture. Liu et al. [109] combined a deep CNN and a continuous conditional random field, and attained visually sharper transitions and local details. Laina et al. [111] developed a deep residual network based on the ResNet [6] and achieved higher accuracy than [109, 110]. Semi-supervised [112] and unsupervised learning [113, 114, 115] setups have also been explored for disparity image prediction. For instance, Godard et al. [115] formulated disparity estimation as an image reconstruction problem, where neural networks were trained to warp left images to match the right. Mancini et al. [116] proposed a CNN that took both RGB images and optical flow images as input to predict distance.

**Self-supervised Frameworks**  Most learning-based work relied on pixel-level ground truth depth training. However, ground truth depth is generally not available and cannot be manually annotated. To address such difficulties, recent focus has shifted towards seeking other supervision signals for training. For instance, Zhou et al. [113] developed an unsupervised learning framework for simultaneous estimation of depth and ego-motion from a monocular camera, using photometric loss as a supervision. However, the depth estimation is only up-to-scale. Mahjourian et al. [117] improved the accuracy by using 3D geometric constraints, and Yin and Shi [118] extended the framework for optical flow estimation. Li et al. [119] recovered the absolute scale by using stereo image pairs.

**Limitations**  However, the accuracy and reliability of the above methods is still far from being practical, especially for robotic tasks that demand high precision and robustness. The state-of-the-art learning-based methods produce an average error (measured by the root mean squared error) of over 50cm in indoor scenarios (e.g., on the NYU Depth Dataset [30]). Such methods perform even worse outdoors, with at least 4 meters of average error on Make3D [120] and KITTI datasets [121].

### 1.4.4 Structure from Motion and SLAM

The geometric structure of a scene can be recovered by using triangulation from different views, when the scene contains sufficient textures or features and the environment is static. A survey for the current state of SLAM is presented in [122]. However, in many practical applications, the sensor might be static with no motion, or there exist transient and dynamic objects. Therefore, structure from motion might not always be applicable.

The idea of leveraging priors on the structure of the environment to improve or enable geometry estimation has been investigated in early work in computer vision for single-
view 3D reconstruction and feature matching [123, 124]. Early work by [125] addresses Structure from Motion by assuming the environment to be piecewise planar. More recently, [126] propose an approach to speed-up stereo reconstruction by computing the disparity at a small set of pixels and considering the environment to be piecewise planar elsewhere. [127] combine live dense reconstruction with shape-priors-based 3D tracking and reconstruction. [128] propose a regularization based on the structure tensor to better capture the local geometry of images. [99] produce high-resolution depth maps from subsampled depth measurements by using segmentation based on both RGB images and depth samples. [129] compute a dense depth map from a sparse point cloud.

1.4.5 Color Image Inpainting

The problem we address also shares some similarities with the image denoising problem as well as the inpainting problem [130, 131] (where the goal is to fill in missing pixels in a color image). Some of the work include on exemplar-based inpainting [132], diffusion methods [133], total variation minimization [134, 135], and deep neural networks [136, 137]. However, the difference in sensor modalities (RGB vs. depth) results in opportunities to better explore the underlying structure of depth images. However, RGB images have a different underlying distribution than depth images, and thus color inpainting is not the same problem as depth completion. Specifically, textures are independent of geometric structures of objects, so a simple planar objects can display colorful patterns whilst a complex shape might have constant colors. Therefore, a naive application of algorithms designed for color images will not produce optimal accuracy for depth estimation.

1.5 Statement of Contributions

This thesis addresses the following question: is it possible to reconstruct a complete depth signal from sparse and incomplete depth samples? In particular, we study the depth reconstruction problem from both the algorithmic and computational perspectives, and develop algorithms for the depth completion problem especially when the depth input is highly sparse. Numerous contributions arise from this study.

The first contribution is an algorithm for efficient reconstruction of 3D planar surfaces. This algorithm assumes that the 3D structure is piecewise-planar, and thus the second-order derivatives of the depth image are sparse. We develop a linear programming problem for recovery of the 3D surfaces under such assumptions, and provide conditions under which the reconstruction is exact. This method requires no learning, but still outperforms deep-learning-based methods under certain conditions.

The second contribution is a deep regression network and a self-supervised learning framework. We formulate the depth completion problem as a pixel-level regression problem and solve it by training a neural network. Additionally, to address the difficulty in gathering ground truth annotations for depth data, we develop a self-supervised framework that trains the regression network by enforcing temporal photometric consistency, using only raw RGB and sparse depth data. The supervised method achieves state-of-the-art accuracy, and the self-supervised approach attains a lower but comparable accuracy.
Our third contribution is a two-stage algorithm for a broad class of inverse problems (e.g., depth completion and image inpainting). We assume that the target image is the output of a generative neural network, and only a subset of the output pixels is observed. The goal is to reconstruct the unseen pixels based on the partial samples. Our proposed algorithm first recovers the corresponding low-dimensional input latent vector using simple gradient-descent, and then reconstructs the entire output with a single forward pass. We provide conditions under which the proposed algorithm achieves exact reconstruction, and empirically demonstrate the effectiveness of such algorithms on real data.

1.6 Organization

This dissertation consists of three contributed algorithms for solving the single-view depth estimation problem. There is a wide spectrum of possible angles to attack the problem, ranging from pure model-based approach to completely data-driven approach. In this dissertation, we provide three different approaches on the spectrum, where each algorithm is based on a different set of modeling assumptions.

In particular, Chapter 2 presents our first contributed algorithm for depth reconstruction from sparse measurements. This is a model-based algorithm, which is developed based on the assumption that the scene contains mostly planar surfaces and requires solving a simple linear programming optimization problem. In Chapter 3, we present the second contribution – a data-driven deep regression approach based on neural networks and self-supervised learning. In Chapter 4, we introduce the third algorithm, which lies between the first two contributions in terms of the spectrum of methods. Specifically, it is based on generative network inversion. We assume that the target image is the output of a given generative neural network, and the missing pixels can be recovered by inverting the network and finding the low-dimensional latent representation. This algorithm achieves exact reconstruction under our assumptions. At the end, in Chapter 5, we provide a summary of the results presented in this dissertation and outline future research directions.
Chapter 2

Planar Surface Reconstruction

In this chapter\(^1\), we present our first set of contributions – a collection of novel algorithms (and the corresponding theoretical foundations) to reconstruct a depth signal (i.e., a laser scan in 2D, or a depth image in 3D, see Figure 2-1) from sparse and incomplete depth measurements. These algorithms are pure model-based, where we make strong assumptions on the scene geometry without tuning any parameters based on data.

![Figure 2-1](image-url)

Figure 2-1: We show how to reconstruct an unknown depth signal (a) from a handful of samples (b). Our reconstruction is shown in (c). Our results also apply to traditional stereo vision and enable accurate reconstruction (f) from few depth measurements (e) corresponding to the edges in the RGB image (d). Figures (a) and (d) are obtained from a ZED stereo camera.

Specifically, we assume that a structured environment (e.g., indoor, urban scenarios) where the depth data exhibits some regularity. For instance, man-made environments are

\(^1\)The work presented in this chapter has been published [7, 91].
characterized by the presence of many planar surfaces and a few edges and corners. This chapter shows how to leverage this regularity to recover a depth signal from a handful of sensor measurements. Our overarching goal is two-fold: to establish theoretical conditions under which depth reconstruction from sparse and incomplete measurements is possible, and to develop practical inference algorithms for depth estimation.

Our first contribution, presented in Section 2.3, is a general formulation of the depth estimation problem. Here we recognize that the “regularity” of a depth signal is captured by a specific function (the $\ell_0$-norm of the 2nd-order differences of the depth signal). We also show that by relaxing the $\ell_0$-norm to the (convex) $\ell_1$-norm, our problem falls within the cosparse model in compressive sensing (CS). We review related work and give preliminaries on CS in Section 2.1 and Section 2.2.

The second contribution, presented in Section 2.4, is the derivation of theoretical conditions for depth recovery. In particular, we provide conditions under which reconstruction of a signal from incomplete measurements is possible, investigate the robustness of depth reconstruction in the presence of noise, and provide bounds on the reconstruction error. Contrary to the existing literature in CS, our conditions are geometric (rather than algebraic) and provide actionable information to guide sampling strategy.

Our third contribution, presented in Section 2.5, is algorithmic. We discuss practical algorithms for depth reconstruction, including different variants of the proposed optimization-based formulation, and solvers that enable fast depth recovery. In particular, we discuss the application of a state-of-the-art solver for non-smooth convex programming, called NESTA [138].

Our fourth contribution, presented in Section 2.6, is an extensive experimental evaluation, including Monte Carlo runs on simulated data and testing with real sensors. The experiments confirm our theoretical findings and show that our depth reconstruction approach is extremely resilient to noise and works well even when the regularity assumptions are partially violated. We discuss many applications for the proposed approach. Besides our motivating scenario of navigation with miniaturized robots, our approach finds application in several endeavors, including data compression and super-resolution depth estimation.

2.1 Related Work on Compressive Sensing

Our work is related to the literature on compressive sensing [139, 140, 141, 142] (CS). While Shannon’s theorem states that to reconstruct a signal (e.g., a depth signal) we need a sampling rate (e.g., the spatial resolution of our sensor) which must be at least twice the maximum frequency of the signal, CS revolutionized signal processing by showing that a signal can be reconstructed from a much smaller set of samples if it is sparse in some domain. CS mainly invokes two principles. First, by inserting randomness in the data acquisition, one can improve reconstruction. Second, one can use $\ell_1$-minimization to encourage sparsity of the reconstructed signal. Since its emergence, CS impacted many research areas, including image processing (e.g., inpainting [143], total variation minimization [144]), data compression and 3D reconstruction [145, 146, 147], tactile sensor data acquisition [148], inverse problems and regularization [149], matrix completion [150], and single-pixel imaging techniques [151, 152, 153]. While most of the CS literature assumes that the original
signal $z$ is sparse in a particular domain, i.e., $z = Dx$ for some matrix $D$ and a sparse vector $x$ (this setup is usually called the *synthesis model*), very recent work considers the case in which the signal becomes sparse after a transformation is applied (i.e., given a matrix $D$, the vector $Dz$ is sparse). The latter setup is called the *analysis (or cosparsity) model* [154, 155]. An important application of the analysis model in compressive sensing is total variation minimization, which is ubiquitous in image processing [144, 156]. In a hindsight we generalize total variation (which applies to piecewise constant signals) to piecewise linear functions.

### 2.2 Preliminaries and Notation

We use uppercase letters for matrices, e.g., $D \in \mathbb{R}^{p \times n}$, and lowercase letters for vectors and scalars, e.g., $z \in \mathbb{R}^n$ and $a \in \mathbb{R}$. Sets are denoted with calligraphic fonts, e.g., $\mathcal{M}$. The cardinality of a set $\mathcal{M}$ is denoted with $|\mathcal{M}|$. For a set $\mathcal{M}$, the symbol $\overline{\mathcal{M}}$ denotes its complement. For a vector $z \in \mathbb{R}^n$ and a set of indices $\mathcal{M} \subseteq \{1, \ldots, n\}$, $z_\mathcal{M}$ is the sub-vector of $z$ corresponding to the entries of $z$ with indices in $\mathcal{M}$. In particular, $z_i$ is the $i$-th entry. The symbols $1$ (resp. $0$) denote a vector of all ones (resp. zeros) of suitable dimension.

The *support set* of a vector is denoted with

$$\text{supp}(z) = \{i \in \{1, \ldots, n\} : z_i \neq 0\}.$$

We denote with $\|z\|_2$ the Euclidean norm and we also use the following norms:

$$\|z\|_\infty \doteq \max_{i=1,\ldots,n} |z_i| \quad (\ell_\infty \text{ norm}) \quad (2.1)$$

$$\|z\|_1 \doteq \sum_{i=1,\ldots,n} |z_i| \quad (\ell_1 \text{ norm}) \quad (2.2)$$

$$\|z\|_0 \doteq |\text{supp}(z)| \quad (\ell_0 \text{ pseudo-norm}) \quad (2.3)$$

Note that $\|z\|_0$ is simply the number of nonzero elements in $z$. The sign vector $\text{sign}(z)$ of $z \in \mathbb{R}^n$ is a vector with entries:

$$\text{sign}(z)_i \doteq \begin{cases} +1 & \text{if } z_i > 0 \\ 0 & \text{if } z_i = 0 \\ -1 & \text{if } z_i < 0 \end{cases}$$

For a matrix $D$ and an index set $\mathcal{M}$, let $D_\mathcal{M}$ denote the sub-matrix of $D$ containing only the rows of $D$ with indices in $\mathcal{M}$; in particular, $D_i$ is the $i$-th row of $D$. Similarly, given two index sets $\mathcal{I}$ and $\mathcal{J}$, let $D_{\mathcal{I},\mathcal{J}}$ denote the sub-matrix of $D$ including only rows in $\mathcal{I}$ and columns in $\mathcal{J}$. Let $I$ denote the identity matrix. Given a matrix $D \in \mathbb{R}^{p \times n}$, we define the following matrix operator norm

$$\|D\|_{\infty \rightarrow \infty} \doteq \max_{i=1,\ldots,p} \|D_i\|_1.$$
In the rest of the chapter we use the *cosparsity model* in CS. In particular, we assume that the signal of interest is sparse under the application of an *analysis operator*. The following definitions formalize this concept.

**Definition 1 (Cosparsity).** A vector $z \in \mathbb{R}^n$ is said to be cosparsely with respect to a matrix $D \in \mathbb{R}^{p \times n}$ if $\|Dz\|_0 \ll p$.

**Definition 2 (D-support and D-cosupport).** Given a vector $z \in \mathbb{R}^n$ and a matrix $D \in \mathbb{R}^{p \times n}$, the $D$-support of $z$ is the set of indices corresponding to the nonzero entries of $Dz$, i.e., $\mathcal{I} = \text{supp}(Dz)$. The $D$-cosupport $\mathcal{J}$ is the complement of $\mathcal{I}$, i.e., the indices of the zero entries of $Dz$.

### 2.3 Problem Formulation

Our goal is to reconstruct 2D depth signals (i.e., a scan from a 2D laser range finder) and 3D depth signals (e.g., a depth image produced by a Kinect or a stereo camera) from partial and incomplete depth measurements. In this section we formalize the depth reconstruction problem, by first considering the 2D and the 3D cases separately, and then reconciling them under a unified framework.

#### 2.3.1 2D Depth Reconstruction

In this section we discuss how to recover a 2D depth signal $z^° \in \mathbb{R}^n$. One can imagine that the vector $z^°$ includes (unknown) depth measurements at discrete angles; this is what a standard planar range finder would measure.

In our problem, due to sensing constraints, we do not have direct access to $z^°$, and we only observe a subset of its entries. In particular, we measure

$$y = Az^° + \eta$$  \hspace{1cm} (sparse measurements)

(2.4)

where the matrix $A \in \mathbb{R}^{m \times n}$ with $m \ll n$ is the *measurement matrix*, and $\eta$ represents measurement noise. The structure of $A$ is formalized in the following definition.

**Definition 3 (Sample set and sparse sampling matrix).** A sample set $\mathcal{M} \subseteq \{1, \ldots, n\}$ is the set of entries of the signal that are measured. A matrix $A \in \mathbb{R}^{m \times n}$ is called a (sparse) sampling matrix (with sample set $\mathcal{M}$), if $A = I_{\mathcal{M}}$.

Recall that $I_{\mathcal{M}}$ is a sub-matrix of the identity matrix, with only rows of indices in $\mathcal{M}$. It follows that $Az = z|_{\mathcal{M}}$, i.e., the matrix $A$ selects a subset of entries from $z$. Since $m \ll n$, we have much fewer measurements than unknowns. Consequently, $z^°$ cannot be recovered from $y$, without further assumptions.

In this chapter we assume that the signal $z^°$ is sufficiently regular, in the sense that it contains only a few “corners”, e.g., Figure 2-2(a). Corners are produced by changes of slope: considering 3 consecutive points at coordinates $(x_{i-1}, z_{i-1}), (x_i, z_i)$, and $(x_{i+1}, z_{i+1})$, \(^2\)

\(^2\)Note that $x$ corresponds to the horizontal axis in Figure 2-2(a), while the depth $z$ is shown on the vertical axis in the figure.

36
there is a corner at $i$ if

$$\frac{z_{i+1} - z_i}{x_{i+1} - x_i} - \frac{z_i - z_{i-1}}{x_i - x_{i-1}} \neq 0. \tag{2.5}$$

In the following we assume that $x_i - x_{i-1} = 1$ for all $i$: this comes without loss of generality since the full signal is unknown and we can reconstruct it at arbitrary resolution (i.e., at arbitrary $x$); hence (2.5) simplifies to $z_{i-1} - 2z_i + z_{i+1} \neq 0$. We formalize the definition of “corner” as follows.

**Definition 4 (Corner set).** Given a 2D depth signal $z \in \mathbb{R}^n$, the corner set $\mathcal{C} \subseteq \{2, \ldots, n-1\}$ is the set of indices $i$ such that $z_{i-1} - 2z_i + z_{i+1} \neq 0$.

Intuitively, $z_{i-1} - 2z_i + z_{i+1}$ is the discrete equivalent of the 2nd-order derivative at $z_i$. We call $z_{i-1} - 2z_i + z_{i+1}$ the **curvature** at sample $i$: if this quantity is zero, the neighborhood of $i$ is flat (the three points are collinear); if it is negative, the curve is locally concave; if it is positive, it is locally convex. To make notation more compact, we introduce the 2nd-order difference operator:

$$D = \begin{bmatrix} 1 & -2 & 1 & 0 & \ldots & 0 \\ 0 & 1 & -2 & 1 & \ldots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & 1 & -2 & 1 \end{bmatrix} \in \mathbb{R}^{(n-2) \times n} \tag{2.6}$$

Then a signal with only a few corners is one where $Dz^\circ$ is sparse. In fact, the $\ell_0$-norm of $Dz^\circ$ counts exactly the number of corners of a signal:

$$\|Dz^\circ\|_0 = |\mathcal{C}| \quad (\# \text{ of corners}) \tag{2.7}$$

where $|\mathcal{C}|$ is the number of corners in the signal.

When operating in indoor environments, it is reasonable to assume that $z^\circ$ has only a few corners. Therefore, we want to exploit this regularity assumption and the partial measurements $y$ in (2.4) to reconstruct $z^\circ$. Let us start from the noiseless case in which $\eta = 0$ in (2.4). In this case, a reasonable way to reconstruct the signal $z^\circ$ is to solve the
following optimization problem:

\[
\min_z \|Dz\|_0 \quad \text{s.t.} \quad Az = y
\]  

(2.8)

which seeks the signal \( z \) that is consistent with the measurements (2.4) and contains the smallest number of corners. Unfortunately, problem (2.8) is NP-hard due to the nonconvexity of the \( \ell_0 \) (pseudo) norm. In this work we study the following relaxation of problem (2.8):

\[
\min_z \|Dz\|_1 \quad \text{s.t.} \quad Az = y
\]  

(2.9)

which is a convex program (it can be indeed rephrased as a linear program), and can be solved efficiently in practice. Section 2.4 provides conditions under which (2.9) recovers the solution of (2.8). Problem (2.9) falls in the class of the cosparse models in CS [155].

In the presence of bounded measurement noise (2.4), i.e., \( \|\eta\|_\infty \leq \varepsilon \), the \( \ell_1 \)-minimization problem becomes:

\[
\min_z \|Dz\|_1 \quad \text{s.t.} \quad \|Az - y\|_\infty \leq \varepsilon
\]  

(2.10)

Note that we assume that the \( \ell_\infty \) norm of the noise \( \eta \) is bounded, since this naturally reflects the sensor model in our robotic applications (i.e., bounded error in each laser beam). On the other hand, most CS literature considers the \( \ell_2 \) norm of the error to be bounded and thus obtains an optimization problem with the \( \ell_2 \) norm in the constraint. The use of the \( \ell_\infty \) norm as a constraint in (2.10) resembles the Dantzig selector of Candès and Tao [157], with the main difference being the presence of the matrix \( D \) in the objective.

### 2.3.2 3D Depth Reconstruction

In this section we discuss how to recover a 3D depth signal \( Z^\circ \in \mathbb{R}^{r \times c} \) (a depth map, as the one in Figure 2-1(a)), using incomplete measurements. As in the 2D setup, we do not have direct access to \( Z^\circ \), but instead only have access to \( m \ll r \times c \) point-wise measurements in the form:

\[
y_{i,j} = Z^\circ_{i,j} + \eta_{i,j}
\]  

(2.11)

where \( \eta_{i,j} \in \mathbb{R} \) represents measurement noise. Each measurement is a noisy sample of the depth of \( Z^\circ \) at pixel \((i, j)\).

We assume that \( Z^\circ \) is sufficiently regular, which intuitively means that the depth signal contains mostly planar regions and only a few “edges”. We define the edges as follows.

**Definition 5 (Edge set).** Given a 3D signal \( Z \in \mathbb{R}^{r \times c} \), the vertical edge set \( \mathcal{E}_V \subseteq \{2, \ldots, r-1\} \times \{1, \ldots, c\} \) is the set of indices \((i, j)\) such that \( Z_{i-1,j} - 2Z_{i,j} + Z_{i+1,j} \neq 0 \). The horizontal edge set \( \mathcal{E}_H \subseteq \{1, \ldots, r\} \times \{2, \ldots, c-1\} \) is the set of indices \((i, j)\) such that \( Z_{i,j-1} - 2Z_{i,j} + Z_{i,j+1} \neq 0 \). The edge set \( \mathcal{E} \) is the union of the two sets: \( \mathcal{E} = \mathcal{E}_V \cup \mathcal{E}_H \).

Intuitively, \((i, j)\) is not in the edge set \( \mathcal{E} \) if the \( 3 \times 3 \) patch centered at \((i, j)\) is planar, while \((i, j) \in \mathcal{E} \) otherwise. As in the 2D case we introduce 2nd-order difference operators \( D_V \) and \( D_H \) to compute the vertical differences \( Z_{i,j-1} - 2Z_{i,j} + Z_{i,j+1} \) and the horizontal
differences $Z_{i-1,j} - 2Z_{i,j} + Z_{i+1,j}$:

\[
D_V Z^\circ \in \mathbb{R}^{(r-2) \times c}, \quad Z^\circ D_H^T \in \mathbb{R}^{r \times (c-2)}
\]  
(2.12)

where the matrices $D_V$ and $D_H$ are the same as the one defined (2.6), but with suitable dimensions; each entry of the matrix $D_V Z^\circ$ contains the vertical (2nd-order) differences at a pixel, while $Z^\circ D_H^T$ collects the horizontal differences.

Following the same reasoning of the 2D case, we obtain the following $\ell_1$-norm minimization

\[
\min_z \quad \| \text{vec}(D_V Z) \|_1 + \| \text{vec}(Z D_H^T) \|_1
\]
\[\text{s.t.} \quad Z_{i,j} = y_{i,j} \quad \text{for each measured pixel } (i, j)\]
(2.13)

where $\text{vec}(\cdot)$ denotes the (column-wise) vectorization of a matrix, and we assume noiseless measurements. In the presence of measurement noise, the equality constraint in (2.13) is again replaced by $|Z_{i,j} - y_{i,j}| \leq \varepsilon$, $\forall (i, j)$, where $\varepsilon$ is an upper bound on the pixel-wise noise $\eta_{i,j}$.

### 2.3.3 Reconciling 2D and 3D Depth Reconstruction

In this section we show that the 3D depth reconstruction problem (2.13) can be reformulated to be closer to its 2D counterpart (2.9), if we vectorize the depth signal (matrix $Z$). For a given signal $Z \in \mathbb{R}^{r \times c}$, we define the number of pixels $n \doteq r \times c$, and we call $z$ the vectorized version of $Z$, i.e., $z \doteq \text{vec}(Z) \in \mathbb{R}^n$. Using standard properties of the vectorization operator, we get

\[
\text{vec}(D_V Z) = (I_c \otimes D_V) z
\]
\[
\text{vec}(Z D_H^T) = (D_H \otimes I_r) z
\]
(2.14)

\[
Z_{i,j} = \text{vec}(e_i^T Z e_j^T) = (e_i^T \otimes e_j^T) z
\]

where $\otimes$ is the Kronecker product, $I_r$ is an identity matrix of size $r \times r$, and $e_i$ is a vector which is zero everywhere except the $i$-th entry which is 1. Stacking all measurements (2.11) in a vector $y \in \mathbb{R}^m$ and using (2.14), problem (2.13) can be written succinctly as follows:

\[
\min_z \quad \| \Delta z \|_1 \quad \text{s.t. } Ax = y
\]
(2.15)

where the matrix $A \in \mathbb{R}^{m \times n}$ (stacking rows in the form $e_i^T \otimes e_j^T$) has the same structure of the sampling matrix introduced in Definition 3, and the “regularization” matrix $\Delta$ is:

\[
\Delta \doteq \begin{bmatrix} I_c \otimes D_V \\ D_H \otimes I_r \end{bmatrix} \in \mathbb{R}^{2(n-r-c) \times n}
\]
(2.16)

Note that (2.15) is the same as (2.9), except for the fact that the matrix $D$ in the objective is replaced with a larger matrix $\Delta$. It is worth noticing that the matrix $\Delta$ is also sparse, with only 3 non-zero entries ($1$, $-2$, and $1$) on each row in suitable (but not necessarily
consecutive) positions.

In the presence of noise, we define an error vector \( \eta \in \mathbb{R}^m \) which stacks the noise terms in (2.11) for each pixel \((i,j)\), and assume pixel-wise bounded noise \( \|\eta\|_\infty \leq \varepsilon \). The noisy 3D depth reconstruction problem then becomes:

\[
\min_{\varepsilon} \|\Delta z\|_1 \quad \text{s.t.} \quad \|Az - y\|_\infty \leq \varepsilon
\]  

(2.17)

Again, comparing (2.10) and (2.17), it is clear that in 2D and 3D we solve the same optimization problem, with the only difference lying in the matrices \( D \) and \( \Delta \).

### 2.4 Analysis: Conditions for Exact Recovery and Error Bounds

This section provides a comprehensive analysis on the quality of the depth signals reconstructed by solving problems (2.9) and (2.10) in the 2D case, and problems (2.15) and (2.17) in 3D. A summary of the key technical results presented in this chapter is given in Table 2.1.

In particular, Section 2.4.1 discusses exact recovery and provides the conditions on the depth measurements such that the full depth signal can be recovered exactly. Since these conditions are quite restrictive in practice (although we will discuss an interesting application to data compression in Section 2.6), Section 2.4.2 analyzes the reconstructed signals under more general conditions. More specifically, we derive error bounds that quantify the distance between the ground truth depth signal and our reconstruction. Section 2.4.3 extends these error bounds to the case in which the depth measurements are noisy.

#### 2.4.1 Sufficient Conditions for Exact Recovery

In this section we provide sufficient conditions under which the full depth signal can be reconstructed exactly from the given depth samples.

Recent results on cosparsity in compressive sensing provide sufficient conditions for exact recovery of a cosparse signal \( z^\diamond \), from measurements \( y = Az^\diamond \) (where \( A \) is a generic matrix). We recall this condition in Proposition 6 below and, after presenting the result, we discuss why this condition is not directly amenable for roboticists to use.

**Proposition 6 (Exact Recovery [154]).** Consider a vector \( z^\diamond \in \mathbb{R}^n \) with \( D \)-support \( \mathcal{I} \) and \( D \)-cosupport \( \mathcal{J} \). Define \( \bar{m} = n - m \). Let \( N \in \mathbb{R}^{\bar{m} \times n} \) be a matrix whose rows span the null space of the matrix \( A \). Let \((\cdot)^\dagger\) denote the Moore-Penrose pseudoinverse of a matrix. If the following condition holds:

\[
C_{er} = \|((N(D_J)^T)^\dagger N(D_I)^T)\|_{\infty \rightarrow \infty} < 1
\]  

(2.18)

then problem (2.9) recovers \( z^\diamond \) exactly.

Despite its generality, Proposition 6 provides only an algebraic condition. In our depth estimation problem, it would be more desirable to have geometric conditions, which sug-
<table>
<thead>
<tr>
<th>2D/3D</th>
<th>Sampling Strategy</th>
<th>Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D &amp; 3D</td>
<td>noiseless</td>
<td>Proposition 6</td>
<td>sufficient condition for exact recovery (algebraic condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noiseless, corners &amp; neighbors</td>
<td>Proposition 7</td>
<td>sufficient condition for exact recovery (geometric condition)</td>
</tr>
<tr>
<td>3D</td>
<td>noiseless, edges &amp; neighbors</td>
<td>Proposition 8</td>
<td>sufficient condition for exact recovery (geometric condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noiseless</td>
<td>Proposition 9</td>
<td>necessary and sufficient condition for optimality (algebraic condition)</td>
</tr>
<tr>
<td>3D</td>
<td>noiseless</td>
<td>Corollary 10</td>
<td>necessary and sufficient condition for optimality (algebraic condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noiseless, twin samples &amp; boundaries</td>
<td>Theorem 13</td>
<td>necessary and sufficient condition for optimality (geometric condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noiseless, twin samples &amp; boundaries</td>
<td>Proposition 14</td>
<td>reconstruction error bound</td>
</tr>
<tr>
<td>3D</td>
<td>noiseless, grid samples</td>
<td>Theorem 17</td>
<td>sufficient condition for optimality (geometric condition)</td>
</tr>
<tr>
<td>3D</td>
<td>noiseless, grid samples</td>
<td>Proposition 18</td>
<td>reconstruction error bound</td>
</tr>
<tr>
<td>2D</td>
<td>noisy</td>
<td>Proposition 19</td>
<td>necessary and sufficient condition for robust optimality (algebraic condition)</td>
</tr>
<tr>
<td>3D</td>
<td>noisy</td>
<td>Corollary 20</td>
<td>necessary and sufficient condition for robust optimality (algebraic condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noisy</td>
<td>Theorem 21</td>
<td>necessary condition for robust optimality (geometric condition)</td>
</tr>
<tr>
<td>2D</td>
<td>noisy, twin samples &amp; boundaries</td>
<td>Proposition 24</td>
<td>reconstruction error bound</td>
</tr>
<tr>
<td>3D</td>
<td>noisy, grid samples</td>
<td>Proposition 25</td>
<td>reconstruction error bound</td>
</tr>
</tbody>
</table>

Table 2.1: Summary of the key theoretical results.
gest the best sampling locations. Our contribution in this section is a geometric interpretation of Proposition 6:

We first provide a result for the 2D case. The proof is given in Appendix A.2.

**Proposition 7 (Exact Recovery of 2D depth signals).** Let \( z^o \in \mathbb{R}^n \) be a 2D depth signal with corner set \( C \). Assuming noiseless measurements (2.4), the following hold:

(i) if the sampling set \( \mathcal{M} \) is the union of the corner set and the first and last entries of \( z^o \), then \( C_{er} = 1 \); 

(ii) if the sampling set \( \mathcal{M} \) includes the corners and their neighbors (adjacent entries), then \( C_{er} = 0 \) and problem (2.9) recovers \( z^o \) exactly.

![Figure 2-3](image)

Figure 2-3: (a) Sampling corners and neighbors in 2D depth estimation guarantees exact recovery of the true signal. (b) Sampling only the corners does not guarantee, in general, that the solution of the \( \ell_1 \)-minimization, namely \( z^* \), coincides with the true signal \( z^o \).

Proposition 7 implies that we can recover the original signal exactly, if we measure the neighborhood of each corner. An example that satisfies such condition is illustrated in Figure 2-3(a). When we sample only the corners, however, Proposition 7 states that \( C_{er} = 1 \); in principle in this case one might still hope to recover the signal \( z^o \), since the condition \( C_{er} < 1 \) in Proposition 6 is only sufficient for exact recovery. But it turns out that in our problem one can find counterexamples with \( C_{er} = 1 \) in which \( \ell_1 \)-minimization fails to recover \( z^o \). A pictorial example is shown in Figure 2-3(b), where we show an optimal solution \( z^* \) which differs from the true signal \( z^o \).

We derive a similar condition for 3D problems. The proof is given in Appendix A.3.

**Proposition 8 (Exact Recovery of 3D depth signals).** Let \( Z^o \) be a 3D depth signal with edge set \( E \). Assume noiseless measurements. If the sampling set \( \mathcal{M} \) includes the edges and theirs (vertical and horizontal) neighbors (adjacent pixels), then \( C_{er} = 0 \), and (2.15) recovers \( \text{vec}(Z^o) \) exactly.

In the experimental section, we show that these initial results already unleash interesting applications. For instance, in stereo vision problems, we could locate the position of the edges from the RGB images and recover the depth in a neighborhood of the edge pixels. Then, the complete depth signal can be recovered (at arbitrary resolution) via (2.15).
2.4.2 Depth Reconstruction from Noiseless Samples

The exact recovery conditions of Proposition 7 and Proposition 8 are quite restrictive if we do not have prior knowledge of the position of the corners or edges. In this section we provide more powerful results that do not require sampling corners or edges. Empirically, we observe that when we do not sample all the edges, the optimization problems (2.9) and (2.15) admit multiple solutions, i.e., multiple signals $z$ attain the same optimal cost. The basic questions addressed in this section are: which signals are in the solution set $S^*$ of problems (2.9) and (2.15)? Is the ground truth signal $z^\circ$ among these optimal solutions? How far can an optimal solution be from the ground truth signal $z^\circ$? In order to answer these questions, in this section we derive optimality conditions for problems (2.9) and (2.15), under the assumption that all measurements are noise-free.

Algebraic Optimality Conditions (noiseless samples)

In this section, we first derive a general algebraic condition for a 2D signal (resp. 3D) to be in the solution set of (2.9) (resp. (2.15)). Then, we translate this algebraic condition into a geometric constraint on the curvature of the signals in the solution set.

**Proposition 9 (2D Optimality).** Let $A$ be the sampling matrix and $M$ be the sample set. Given a signal $z \in \mathbb{R}^n$ which is feasible for (2.9), $z$ is a minimizer of (2.9) if and only if there exists a vector $u$ such that

$$
(D^T)_{\mathcal{M}} u = 0 \text{ and } u_{\mathcal{I}} = \text{sign}(Dz)_{\mathcal{I}} \text{ and } \|u\|_\infty \leq 1 \quad (2.19)
$$

where $\mathcal{I}$ is the $D$-support of $z$ (i.e., the set of indices of the nonzero entries of $Dz$) and $\mathcal{M}$ is the set of entries of $z$ that we do not sample (i.e., the complement of $\mathcal{M}$).

The proof of Proposition 9 is based on the subdifferential of the $\ell_1$-minimization problem and is provided in Appendix A.4. An analogous result holds in 3D.

**Corollary 10 (3D optimality).** A given signal $Z$ is in the set of minimizers of (2.15) if and only if the conditions of Proposition 9 hold, replacing $D$ with $\Delta$ in eqs. (2.19).

We omit the proof of Corollary 10 since it follows the same line of the proof of Proposition 9.

Analysis of 2D Reconstruction (noiseless samples)

In this section we derive necessary and sufficient geometric conditions for $z^\circ$ to be in the solution set of (2.9). Using these findings we obtain two practical results: (i) an upper bound on how far any solution $z^\ast$ of (2.9) can be from the ground truth signal $z^\circ$; (ii) a general algorithm that recovers $z^\circ$ even when the conditions of Proposition 7 fail (the algorithm is presented in Section 2.5.1).

To introduce our results, we need the following definition.
Definition 11 (2D Sign Consistency). Let $s_k = \text{sign}(z_{k-1} - 2z_k + z_{k+1})$ (sign of the curvature at $k$). A 2D depth signal $z$ is sign consistent if, for any two consecutive samples $i < j \in \mathcal{M}$, one of the two conditions holds:

(i) no sign change: for any two integers $k, h$, with $i \leq k, h \leq j$, if $s_k \neq 0$ and $s_h \neq 0$, then $s_k = s_h$;

(ii) sign change only at the boundary: for any integer $k$, with $i < k < j$, $s_k = 0$.

This technical definition has a clear geometric interpretation. In words, a signal $z$ is sign consistent, if its curvature does not change sign (i.e., it is either convex or concave) within each interval between consecutive samples. See Figure 2-4 for examples of sign consistency, alongside with a counter-example.

![Figure 2-4: Examples of sign-consistent (shown in black) and sign-inconsistent (shown in blue) 2D depth signal.](image)

In the following we show that any optimal solutions for problem (2.9) must be sign consistent. In order to simplify the analysis for Theorem 13 below, we assume that we pick pairs of consecutive samples (rather than individual, isolated samples). We formalize this notion as follows.

Definition 12 (Twin samples). A twin sample is a pair of consecutive samples, i.e., $(i, i+1)$ with $i \in \{1, \ldots, n-1\}$.

Theorem 13 (2D Sign Consistency $\iff$ Optimality). Let $z$ be a 2D signal which is feasible for problem (2.9). Assume that the sample set includes only twin samples and we sample the “boundary” of the signal, i.e., $z_1$, and $z_n$. Then, $z$ is optimal for (2.9) if and only if it is sign consistent.

The proof of Theorem 13 is given in Appendix A.5. This theorem provides a tight geometric condition for a signal to be optimal. More specifically, a signal is optimal for problem (2.9) if it passes through the given set of samples (i.e., it satisfies the constraint in (2.9)) and does not change curvature between consecutive samples. This result also provides insights into the conditions under which the ground truth signal will be among the minimizers of (2.9), and how one can bound the depth estimation error, as stated in the following proposition.
Proposition 14 (2D Recovery Error - noiseless samples). Let \( z^\circ \) be the ground truth signal generating noiseless measurements (2.4). Assume that we sample the boundary of \( z^\circ \) and the sample set includes a twin sample in each linear segment in \( z^\circ \). Then, \( z^\circ \) is in the set of minimizers of (2.9). Moreover, denote with \( \tilde{z} \) the naive solution obtained by connecting consecutive samples with a straight line (linear interpolation). Then, any optimal solution \( z^* \) lies between \( z^\circ \) and \( \tilde{z} \), i.e., for any index \( i \in \{1, \ldots, n\} \), it holds \( \min(z^\circ_i, \tilde{z}_i) \leq z^*_i \leq \max(z^\circ_i, \tilde{z}_i) \). Moreover, it holds

\[
\|z^\circ - z^*\|_\infty \leq \max_{i \in \mathcal{M}} d_i \cos(\theta_i)
\]

where \( d_i \) is the distance between the sample \( i \) and the nearest corner in \( z^\circ \), while \( \theta_i \) is the angle that the line connecting \( i \) with the nearest corner forms with the vertical.

Figure 2-5: (a) Region between a pair of twin samples, with twin samples not including corners. \( d_i \) and \( d_j \) represent distances between samples and the intersection of two lines. The up and down arrows represent the change of slopes. (b) Given a set of noiseless twin samples, all possible optimal solutions in 2D are contained in the envelope shown in gray.

A visualization of the parameters \( d_i \) and \( \theta_i \) is given in Figure 2-5(a). The proof of Proposition 14 is given in Appendix A.6.

Proposition 14 provides two important results. First, it states that any optimal solution \( z^* \) (e.g., the dotted green line in Figure 2-2(b)) should lie between the ground truth depth \( z^\circ \) (solid black line) and the naive solution \( \tilde{z} \) (dashed blue line). In other words, any arbitrary set of twin samples defines an envelope that contains all possible solutions. An example of such envelope is illustrated in Figure 2-5(b). The width of this envelope bounds the maximum distance between any optimal solution and the ground truth, and hence such envelope provides a point-wise quantification of the reconstruction error. Second, Proposition 14 provides an upper bound on the overall reconstruction error in eq. (2.20). The inequality implies that the reconstruction error grows with the parameter \( d_i \), the distance between our samples and the corners. In addition, the error also increases if the parameter \( \theta_i \) is small, meaning that the ground truth signals are “pointy” and there exist abrupt changes of slope between consecutive segments. An instance of such “pointy” behavior is the second corner from right in Figure 2-5(b).
We will further show in Section 2.5 that Proposition 14 has algorithmic implications. Based on Proposition 14, we design an algorithm that exactly recovers a 2D signal, even when the sample set does not contain all corners. Before moving to algorithmic aspects, let us consider the 3D case.

**Analysis of 3D Reconstruction (noiseless samples)**

In this section we provide a sufficient geometric condition for a 3D signal to be in the solution set of (2.15). We start by introducing a specific sampling strategy (the analogous of the twin samples in 2D) to simplify the analysis.

**Definition 15 (Grid samples and Patches).** Given a 3D signal $Z \in \mathbb{R}^{r \times c}$, a grid sample set includes pairs of consecutive rows and columns of $Z$, along with the boundaries (first and last two rows, first and last two columns). This sampling strategy divides the image in rectangular patches, i.e., sets of non-sampled pixels enclosed by row-samples and column-samples.

![Image](a)

Figure 2-6: (a) Illustration of a grid sample set, along with 6 non-sampled patches in white. (b) A cross section of the envelope in 3D.

Figure 2-6(a) shows an example of grid samples and patches. If we have $K$ patches and we denote the set of non-sampled pixels in patch $i$ with $\overline{M}_i$, then the union $\mathcal{M} \cup \{\overline{M}_i\}_{i=1}^K$ includes all the pixels in the depth image. We can now extend the notion of sign consistency to the 3D case.

**Definition 16 (3D Sign Consistency).** Let $Z \in \mathbb{R}^{r \times c}$ be a 3D depth signal. Let $\mathcal{M}$ be a grid sampling set and $\{\overline{M}_i\}_{i=1}^K$ be the non-sampled patches. Let $Z_{\overline{M}_i}$ be the restriction of $Z$ to its entries in $\overline{M}_i$. Then, $Z$ is called 3D sign consistent if for all $i = \{1, \ldots, K\}$, the nonzero entries of $\text{sign}(\text{vec}(DZ_{\overline{M}_i}))$ are all $+1$ or $-1$, and the nonzero entries of $\text{sign}(\text{vec}(Z_{\overline{M}_i}D^T))$ are all $+1$ or $-1$, where $D$ is the 2nd-order difference operator (2.6) of suitable dimension.
Intuitively, 3D sign consistency indicates that the sign of the signal’s curvature does not change, either horizontally or vertically, within each non-sampled patch. We now present a sufficient condition for \( Z^\circ \) to be in the solution set of (2.15).

**Theorem 17 (3D Sign Consistency \( \Rightarrow \) Optimality).** Let \( Z \in \mathbb{R}^{r \times c} \) be a 3D signal, feasible for problem (2.15). Assume the sample set \( \mathcal{M} \) is a grid sample set. Then \( Z \) is in the set of minimizers of (2.15) if it is 3D sign consistent.

The proof is given in Appendix A.7. Theorem 17 is weaker than Theorem 13, the 2D counterpart, since our definition of 3D sign consistency is only sufficient, but not necessary, for optimality. Nevertheless, it can be used to bound the depth recovery error as follows.

**Proposition 18 (3D Recovery Error - noiseless samples).** Let \( Z^\circ \in \mathbb{R}^{r \times c} \) be the ground truth signal generating noiseless measurements (2.4). Let \( \mathcal{M} \) be a grid sampling set and assume \( Z^\circ \) to be 3D sign consistent with respect to \( \mathcal{M} \). Moreover, let \( Z \in \mathbb{R}^{r \times c} \) and \( \bar{Z} \in \mathbb{R}^{r \times c} \) be the point-wise lower and upper bound of the row-wise envelope, built as in Figure 2-5(b) by considering each row of the 3D depth signal as a 2D signal. Then, \( Z^\circ \) is an optimal solution of (2.15), and any other optimal solution \( Z^* \) of (2.15) satisfies:

\[
|Z^\circ_{i,j} - Z^*_{i,j}| \leq \max(|Z_{i,j} - Z^*_{i,j}|, |\bar{Z}_{i,j} - Z^*_{i,j}|)
\]  

Roughly speaking, if our grid sampling is “fine” enough to capture all changes in the sign of the curvature of \( Z^\circ \), then \( Z^\circ \) is among the solutions of (2.15). Despite the similarity to Proposition 14, the result in Proposition 18 is weaker. More specifically, Proposition 18 is based on the fact that we can compute an envelope only for the ground truth signal (but not for all the optimal solutions, as in Proposition 14). Moreover, the estimation error bound in eq. (2.21) can be only computed a posteriori, i.e., after obtaining an optimal solution \( Z^* \). Nevertheless, the result can be readily used in practical applications, in which one wants to bound the depth estimation error. An example of the row-wise envelope is given in Figure 2-6(b).

### 2.4.3 Depth Reconstruction from Noisy Samples

In this section we analyze the depth reconstruction quality for the case where the measurements (2.4) are noisy. In other words, we now focus on problems (2.10) and (2.17).

**Algebraic Optimality Conditions (noisy samples)**

In this section, we derive a general algebraic condition for a 2D signal (resp. 3D) to be in the solution set of (2.10) (resp. (2.17)). This condition generalizes the optimality condition of Section 2.4.2 to the noisy case. In Section 2.4.3 and Section 2.4.3, we apply this algebraic condition to bound the depth reconstruction error.

**Proposition 19 (2D robust optimality).** Let \( A \) be the sampling matrix, \( \mathcal{M} \) be the sample set and \( y \) be the noisy measurements as in (2.4), with \( \|\eta\|_\infty \leq \varepsilon \) and \( \varepsilon > 0 \). Given a signal
which is feasible for (2.10), define the active set $\mathcal{A} \subset \mathcal{M}$ as follows

$$\mathcal{A} = \{ i \in \mathcal{M} : |y_i - z_i| = \varepsilon \}.$$  

(2.22)

We also define its two subsets

$$\mathcal{A}_\uparrow = \{ i \in \mathcal{M} : y_i - z_i = \varepsilon \},$$

$$\mathcal{A}_\downarrow = \{ i \in \mathcal{M} : y_i - z_i = -\varepsilon \}.$$  

(2.23)

Also denote $\bar{\mathcal{A}} = \mathcal{M} \setminus \mathcal{A}$. Then $z$ is a minimizer of (2.10) if and only if there exists a vector $u$ such that

$$\begin{align*}
(D^T)_{\mathcal{M} \cup \bar{\mathcal{A}}} u &= 0 \quad \text{and} \quad u_\mathcal{I} = \text{sign}(Dz)_\mathcal{I} \quad \text{and} \quad \|u\|_\infty \leq 1 \quad \text{(2.24)} \\
(D^T)_{\mathcal{A}_\uparrow} u &\geq 0 \quad \text{and} \quad (D^T)_{\mathcal{A}_\downarrow} u \leq 0 \quad \text{(2.25)}
\end{align*}$$

where $\mathcal{I}$ is the $D$-support of $z$, and $\bar{\mathcal{M}}$ is the set of un-sampled entries in $z$ (i.e., the complement of $\mathcal{M}$).

Figure 2-7: A illustration of the active set $\mathcal{A}$ of samples in Proposition 19. The three corners, as well as the second sample from the right, are all within the active set. A measurement $y_i$ is active if the reconstructed signal $z$ hits the boundary of $y_i$’s associated error bound.

The proof is given in Appendix A.9. A visual illustration of the active set is given in Figure 2-7. We will provide some geometric insights on the algebraic conditions in Proposition 19 in the next two sections. Before moving on, we re-ensure that the robust optimality conditions straightforwardly extends to the 3D case.

**Corollary 20 (3D robust optimality).** A given signal $Z$ is in the set of minimizers of (2.17) if and only if the conditions of Proposition 19 hold, replacing $\Delta$ with $D$ in eqs. (2.24)-(2.25).

We skip the proof of Corollary 20 since it proceeds along the same line of the proof of Proposition 19.
Analysis of 2D Reconstruction (noisy samples)

In this section we consider the 2D case and provide a geometric interpretation of the algebraic conditions in Proposition 19. The geometric interpretation follows from a basic observation, which enables us to relate the noisy case with our noiseless analysis of Section 2.4.2. The observation is that if a signal satisfies the robust optimality conditions (2.24)-(2.25) then it also satisfies the noiseless optimality condition (2.19), hence being sign consistent, as per Theorem 13.

**Theorem 21 (Robust optimality ⇒ 2D Sign Consistent).** Let \( z^* \) be a 2D signal which is optimal for problem (2.10), and assume that the sample set includes only twin samples and we sample the “boundary” of the signal, i.e., \( z_1 \) and \( z_n \). Then, \( z^* \) is 2D sign consistent.

We present a brief proof for Theorem 21 below.

**Proof.** Note that \( (D^T)_{\mathcal{M} \cup \mathcal{A}} u = 0 \) \( \Rightarrow (D^T)_{\mathcal{M}} u = 0 \). In other words, condition (2.24) implies condition (2.19), which in turn is equivalent to sign consistency as per Theorem 13. Therefore, we come to the conclusion that any optimal solution of (2.10) must be sign consistent. \( \square \)

Theorem 21 will help establish error bounds on the depth reconstruction. Before presenting these bounds, we formally define the 2D sign consistent \( \epsilon \)-envelope.

**Definition 22 (2D Sign Consistent \( \epsilon \)-envelope).** Assume that the sample set includes only twin samples and we sample the “boundary” of the signal, i.e., \( z_1 \) and \( z_n \). Moreover, for each pair of consecutive twin samples \( i, i+1 \) and \( j, j+1 \), define the following line segments for \( k \in (i + 1, j) \):

\[
\begin{align*}
(1) \quad z_k - (y_i - \epsilon) &= \frac{(y_{i+1} + \epsilon) - (y_i - \epsilon)}{x_{i+1} - x_i} \\
(2) \quad z_k - (y_i + \epsilon) &= \frac{(y_{i+1} - \epsilon) - (y_i - \epsilon)}{x_{i+1} - x_i} \\
(3) \quad z_k - (y_{i+1} + \epsilon) &= \frac{(y_j + \epsilon) - (y_{j+1} + \epsilon)}{x_j - x_{j+1}} \\
(4) \quad z_k - (y_{i+1} - \epsilon) &= \frac{(y_j - \epsilon) - (y_{j+1} - \epsilon)}{x_j - x_{j+1}} \\
(5) \quad z_k - (y_i + \epsilon) &= \frac{(y_{i+1} + \epsilon) - (y_i - \epsilon)}{x_i - x_{i+1}} \\
(6) \quad z_k - (y_{i+1} - \epsilon) &= \frac{(y_{i+1} - \epsilon) - (y_i + \epsilon)}{x_{i+1} - x_j} 
\end{align*}
\]

Further define the following signals:

\[
\bar{z} := \begin{cases} 
\max \{(1), (3), (5)\}, & \text{if (1) and (3) intersect} \\
(5), & \text{otherwise}
\end{cases}
\]

and

\[
\bar{z} := \begin{cases} 
\max \{(2), (4), (6)\}, & \text{if (2) and (4) intersect} \\
(6), & \text{otherwise}
\end{cases}
\]
where \( \max\{(1), (3), (5)\} \) denotes the point-wise maximum among the segments in eqs. (1), (3), and (5). We define the 2D sign consistent \( \epsilon \)-envelope as the region enclosed between the upper bound \( \bar{z} \in \mathbb{R}^n \) and the lower bound \( \underline{z} \in \mathbb{R}^n \).

A pictorial representation of the line segments (1)-(6) in Definition 22 is given in Figure 2-8(a)-(b). Figure 2-8(a) shows an example where line segment (1) intersects with (3) and line segment (2) intersects with (4). In Figure 2-8(b), these line segments do no intersect. An example of the resulting 2D sign consistent \( \epsilon \)-envelope is illustrated in Figure 2-8(c).

Our interest towards the 2D sign consistent \( \epsilon \)-envelope is motivated by the following proposition.

**Proposition 23 (2D Sign Consistent \( \epsilon \)-envelope).** Under the conditions of Definition 22, any 2D sign-consistent signal, belongs to the 2D sign consistent \( \epsilon \)-envelope.

![Figure 2-8](image)

Figure 2-8: (a) Illustration of the line segments (1)-(6) in Definition 22 with intersection. (b) Line segments (1)-(6) without intersection. (c) An example of 2D sign consistent \( \epsilon \)-envelope.

The proof of Proposition 23 is given in Appendix A.10.

Next we introduce a proposition that characterizes the depth reconstruction error bounds of an optimal solution.

**Proposition 24 (2D Recovery Error - noisy samples).** Let \( z^\circ \in \mathbb{R}^n \) be the ground truth generating noisy measurements (2.4). Assume that we sample the boundary of \( z^\circ \) and the sample set includes a twin sample in each linear segment in \( z^\circ \). Then, \( z^\circ \) belongs to the 2D sign consistent \( \epsilon \)-envelope, and any optimal solution \( z^* \) of (2.10) also lies in the \( \epsilon \)-envelope. Moreover, denoting with \( \underline{z} \in \mathbb{R}^n \) and \( \bar{z} \in \mathbb{R}^n \), the point-wise lower and upper bound of the \( \epsilon \)-envelope (Definition 22), and considering any consecutive pairs of twin samples \( i, i + 1 \) and \( j, j + 1 \), for all \( k \in (i + 1, j) \), it holds:

\[
|z^\circ_k - z^*_k| \leq \bar{z}_k - \underline{z}_k.
\]

The proof of Proposition 24 is given in Appendix A.11.
Analysis of 3D Reconstruction (noisy samples)

In this section we characterize the error bounds of an optimal solution $Z^\star$ of (2.17) in the noisy case. The result is similar to its noiseless counterpart in Proposition 18.

Proposition 25 (3D Recovery Error - noisy samples). Let $Z^\circ \in \mathbb{R}^{r \times c}$ be the ground truth generating noisy measurements (2.4). Let $\mathcal{M}$ be a grid sample set and assume $Z^\circ$ to be 3D sign consistent with respect to $\mathcal{M}$. Moreover, let $Z \in \mathbb{R}^{r \times c}$ and $\bar{Z} \in \mathbb{R}^{r \times c}$ be the point-wise lower and upper bound of the row-wise 2D sign consistent $\epsilon$-envelope, built as in Figure 2-8(b) by considering each row of the 3D depth signal as a 2D signal. Then, given any optimal solution $Z^\star$ of (2.17), it holds that

$$|Z_{i,j}^\circ - Z^\star_{i,j}| \leq \max( |Z_{i,j} - Z^\star_{i,j}|, |\bar{Z}_{i,j} - Z^\star_{i,j}|) \quad (2.26)$$

The proof for Proposition 25 follows the same line as the proof of Proposition 18, and we omit the proof for brevity.

2.5 Algorithms and Fast Solvers

The formulations discussed so far, namely (2.9), (2.10), (2.15), (2.17), directly translate into algorithms: each optimization problem can be solved using standard convex programming routines and returns an optimal depth signal.

This section describes two algorithmic variants that further enhance the quality of the depth reconstruction (Section 2.5.1), and then presents a fast solver for the resulting $\ell_1$-minimization problems (Section 2.5.2).

```plaintext
input : Measurements $y$, and sample set $\mathcal{M}$, including boundary and twin samples
output: Original signal $z^\circ$
/* solve $\ell_1$-minimization */
1 create matrices $A$ (Definition 3) and $D$ (eq. (2.6)) ;
2 solve $(f^\star, z^\star) = \min_z \|Dz\|_1 \quad \text{s.t. } Az = y ;$
/* populate a vector of signs $s \in \{-1,0,+1\}^n$ */
3 for consecutive twin samples $(i - 1, i), (j, j + 1)$ do
4     foreach $k \in \{i + 1, \ldots, j - 1\}$ do
5         set $s_k = \text{sign}((z^\star_{j+1} - z^\star_j) - (z^\star_i - z^\star_{i-1}))$
6     end
7 end
/* recover $z^\circ$ within the solution set */
8 $z^\circ = \text{argmin}_z s^T z \quad \text{s.t. } Az = y \text{ and } \|Dz\|_1 \leq f^\star ;$
9 return $z^\circ$.
```

**Algorithm 1:** Exact recovery of 2D depth signals.
2.5.1 Enhanced Recovery in 2D and 3D

In this section we describe other algorithmic variants for the 2D and 3D case. Section 2.5.1 proposes a first algorithm that solves 2D problems and is inspired by Proposition 14. Section 2.5.1 discusses variants of (2.15) for 3D problems.

Enhanced Recovery in 2D problems

Proposition 14 dictates that any optimal solution of (2.9) lies between the naive interpolation solution and the ground truth signal $z^\circ$ (recall Figure 2-2(b)). Algorithm 1 is based on a simple idea: on the one hand, if the true signal is concave between two consecutive samples (cf. with the first corner in Figure 2-2(b)), then we should look for an optimal signal having depth “as large as possible” in that particular interval (while still being within the optimal set of (2.9)); on the other hand, if the shape is convex (second corner in Figure 2-2(b)) we should look for an optimal signal with depth as “as small as possible”, since this is the closest to $z^\circ$.

Algorithm 1 first solves problem (2.9) and computes an optimal solution $z^\star$ and the corresponding optimal cost $f^\star$ (lines 1-2). Let us skip lines 3-5 for the moment and take a look at line 8: the constraints in this optimization problem include the same constraint of line 2 ($Az = y$), plus an additional constraint in line 2 ($\|Dz\|_1 \leq f^\star$) that restricts $z$ to stay within the optimal solution set of (2.9). Therefore, it only remains to design a new objective function that “encourages” a solution that is close to $z^\circ$ while still being within this optimal set. To this end, we use a simple linear objective $s^T z$, where $s \in \{0, \pm 1\}^n$ is a vector of coefficients, such that the objective function penalizes large entries in the signal $z$ if $s_k = +1$, and rewards large entries when $s_k = -1$. More specifically, the procedure for choosing a proper coefficient $s_k$ is as follows. For any consecutive pairs of twin samples $(i-1, i)$ and $(j, j+1)$, the algorithm looks at the slope difference between the second pair (i.e., $z^\star_{j+1} - z^\star_j$) and the first pair ($z^\star_i - z^\star_{i-1}$). If this difference is negative, then the function $z^\star$ is expected to be concave between the samples. In this case the sign $s_k$ for any point $k$ between the samples is set to $-1$. If the difference is positive, then the signs are set to $+1$. Otherwise the signs will be 0. We prove the following result.

**Corollary 26 (Exact Recovery of 2D signals by Algorithm 1).** Under the assumptions of Proposition 14, Algorithm 1 recovers the 2D depth signal $z^\circ$ exactly.

The proof is in Appendix A.12. Although Algorithm 1 is designed for noiseless samples, in the experiments in Section 2.6.1 we also test a noisy variant by substituting the constraints in lines 2 and 8 with $\|Az - y\|_\infty \leq \varepsilon$.

Enhanced Recovery in 3D problems

In the formulations (2.15) and (2.17) we used the matrix $\Delta$ to encourage “flatness”, or in other words, regularity of the depth signals. In this section we discuss alternative objective functions which we evaluate experimentally in Section 2.6. These objectives simply adopt different definitions for the matrix $\Delta$ in (2.15) and (2.17). For clarify, we denote the formulation introduced earlier in this chapter (using the matrix $\Delta$ defined in (2.16)) as the
“L1” formulation (also recalled below), and we introduce two new formulations, denoted as “L1diag” and “L1cart”, which use different objectives.

**L1 formulation:** Although we already discussed the structure of the matrix $\Delta$ in Section 2.3.3, here we adopt a slightly different perspective that will make the presentation of the variants L1diag and L1cart clearer. In particular, rather than taking a matrix view as done in Section 2.3.3, we interpret the action of the matrices $D_V$ and $D_H$ in eq. (2.13) as the application of a kernel (or convolution filter) to the 3D depth signal $Z$. In particular, we note that:

$$D_V Z = Z * K_{xx} \quad \quad (Z D_H^T)^T = Z * K_{yy}$$

where “*” denotes the action of a discrete convolution filter and the kernels $K_{xx}$ and $K_{yy}$ are defined as

$$K_{xx} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad K_{yy} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$ 

Intuitively, $K_{xx}$ and $K_{yy}$ applied at a pixel return the 2nd-order differences along the horizontal and vertical directions at that pixel, respectively. The L1 objective, presented in Section 2.3.3, can be then written as:

$$f_{L1}(Z) = \| \text{vec}(Z * K_{xx}) \|_1 + \| \text{vec}(Z * K_{yy}) \|_1.$$ 

**L1diag formulation:** While L1 only penalizes, for each pixel, variations along the horizontal and vertical direction, the objective of the L1diag formulation includes an additional 2nd-order derivative, which penalizes changes along the diagonal direction. This additional term can be written as $\| \text{vec}(Z * K_{xy}) \|_1$, where the kernel $K_{xy}$ is:

$$K_{xy} = \frac{1}{4} \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}.$$ 

Therefore, the objective in the L1diag formulation is

$$f_{L1diag}(Z) = f_{L1}(Z) + \| \text{vec}(Z * K_{xy}) \|_1.$$ 

**L1cart formulation:** When introducing the L1 formulation in Section 2.3, we assumed that we reconstruct the depth at uniformly-spaced point, i.e., the $(x, v)^3$ coordinates of each point belong to a uniform grid; in other words, looking at the notion of curvature in (2.5), we assumed $x_{i+1} - x_i = x_i - x_{i-1} = 1$ (also $v_{i+1} - v_i = v_i - v_{i-1} = 1$ in the 3D case). While this comes without loss of generality, since the full profile is unknown and we can reconstruct it at arbitrary resolution, we note that typical sensors, even in 2D, do not produce measurements with uniform spacing, see Figure 2-9.

For this reason, in this section we generalize the L1 objective to account for irregularly-spaced points. If we denote with $x_{i,j}$ and $v_{i,j}$ the horizontal and vertical coordinates of the 3D point observed at pixel $(i, j)$, a general expression for the horizontal and vertical 2nd-order
Figure 2-9: A toy example illustrating that while a 2D lidar produces measurements at fixed angular resolution, the resulting Cartesian coordinates are not equally spaced, i.e., \( \Delta x_1 \neq \Delta x_2 \). This occurs in both lidars and perspective cameras, hence motivating the introduction of the \( \text{L1} \text{cart} \) formulation.

Order differences is:

\[
Z \ast K_{xx}^{\text{L1-cart}} \quad Z \ast K_{yy}^{\text{L1-cart}}
\]

(2.28)

where the convolution kernels at pixel \((i, j)\) are defined as:

\[
K_{xx}^{\text{cart}}(i, j) = \begin{bmatrix}
0 & \left( \frac{1}{x_{i,j} - x_{i-1,j}} + \frac{1}{x_{i+1,j} - x_{i,j}} \right) & 0 \\
0 & 0 & 0 \\
\frac{1}{x_{i+1,j} - x_{i,j}} & 0 & 0
\end{bmatrix},
\]

(2.29)

\[
K_{yy}^{\text{cart}}(i, j) = \begin{bmatrix}
0 & 0 & 0 \\
\frac{1}{v_{i,j} - v_{i,j-1}} & 0 & 0 \\
0 & \frac{1}{v_{i,j+1} - v_{i,j}} & 0
\end{bmatrix}
\]

The kernels \( K_{xx}^{\text{cart}}(i, j) \) and \( K_{yy}^{\text{cart}}(i, j) \) simplify to \( K_{xx} \) and \( K_{yy} \) when the points are uniformly spaced, and can be used to define a new objective function:

\[
f_{\text{L1cart}}(Z) \doteq \| \text{vec}(Z \ast K_{xx}^{\text{L1-cart}}) \|_1 + \| \text{vec}(Z \ast K_{yy}^{\text{L1-cart}}) \|_1.
\]

\( \text{L1cart} \) may be used to query the depth at arbitrary points and in this sense it is more general than \( \text{L1} \). On the downside, we notice that extra care should be taken to ensure that the denominators in the entries of the kernels \( K_{xx}^{\text{cart}}(i, j) \) and \( K_{yy}^{\text{cart}}(i, j) \) do not vanish, and small denominators (close to zero) may introduce numerical errors. For this reason, in our tests, we add a small positive constant \( \delta \) to all denominators.

### 2.5.2 Fast Solvers

All the formulations presented in this chapter, including the algorithmic variants proposed in Section 2.5.1, rely on solving the optimization problems (2.9), (2.15), (2.17), and (2.10) efficiently. Despite the convexity of these problems, off-the-shelf solvers based on interior point methods tend to be slow and do not scale to very large problems. Recalling that in
the 3D case, the number of unknown variables in our problems is equal to the number of non-sampled pixels in the depth map, these optimization problems can easily involve more than $10^5$ variables. Indeed, in the experiments in Section 2.6.2 we show that off-the-self solvers such as cvx/MOSEK [158, 159] are quite slow and practically unusable for 3D signals larger than $100 \times 100$ pixels.

For these reasons, in this section we discuss a more efficient first-order method to solve these minimization problems. This solver is a variant of NESTA, an algorithm for fast $\ell_1$ minimization recently developed by [138] and based on Nesterov’s method for nonsmooth optimization [160, 161]. We tailor NESTA to our specific optimization problems with $\ell_\infty$-norm constraints, instead of the original $\ell_2$ norm used in [138]. In this section we focus on the 2D problem (2.10), since the algorithm is identical in the 3D case (with the only exception that the matrix $\Delta$ is used in place of $D$).

In this section, we provide an overview of NESTA, adapted to problem (2.10), while we leave technical details to Section A.13. NESTA solves convex optimization problems with nonsmooth objectives, in the general form:

$$\min_z f(z) \quad \text{s.t. } z \in Q$$

where $f(z)$ is a nonsmooth convex function and $Q$ is a convex set. The basic idea in NESTA is to replace the original objective $f(z)$ with a smooth approximation $f_\mu(z)$

$$\min_z f_\mu(z) \quad \text{s.t. } z \in Q$$

where $\mu$ is a parameter controlling the smoothness of $f_\mu(z)$ and such that when $\mu$ goes to zero, $f_\mu(z)$ approaches $f(z)$.

In our problem (2.10), we have $f(z) = \|Dz\|_1$ and $Q = \{z : \|Az - y\|_\infty \leq \varepsilon\}$. Following [160], we first notice that our nonsmooth objective can be written as:

$$f(z) = \|Dz\|_1 = \max_{u: \|u\|_\infty \leq 1} \langle u, Dz \rangle.$$

(2.32)

Then a convenient choice for $f_\mu(z)$ is

$$f_\mu(z) = \max_{u: \|u\|_\infty \leq 1} \langle u, Dz \rangle - \mu \frac{\|u\|_2^2}{2}.$$  

(2.33)

The function $f_\mu(z)$ is differentiable, see [160], and its gradient is Lipschitz with constant $L_\mu$ (Section A.13 provides an explicit expression for the constant $L_\mu$). It can be readily noticed from Equation 2.33 that when $\mu$ goes to zero, $f_\mu(z)$ approaches our objective $f(z)$.

NESTA adopts a continuation approach, in that it solves a sequence of optimization problems with decreasing values of $\mu$, such that the result of the last optimization problem approximates closely the solution of $f(z)$. The advantage in doing so is that, instead of minimizing directly $f(z)$ with nonsmooth optimization techniques which are generally slow, at each iteration NESTA applies Nesterov’s accelerated gradient method to the smooth function $f_\mu(z)$, ensuring an optimal convergence rate of $O(1/K^2)$ in the number of gradient iterations $K$. 

55
**input**: Measurements $y$, sampling matrix $A$, noise level $\varepsilon$, initial guess $z^{(0)}$, desired final smoothing parameter value $\mu_f$, maximum Nesterov’s iterations $K$, continuation steps $T$, stopping criterion $\tau$

**output**: Approximate solution $z^{(K)}$ for Problem 2.10

/* initialize parameters */
1. initialize $\mu_0 = \|D^Ty\|_\infty$;
2. compute $\gamma = (\mu_f - \mu_0)^{1/T}$;

/* outer iterations with decreasing $\mu$ */
3. for $t = 1 : T$ do
4. set $\mu = \mu_{t-1}$;
5. /* Nesterov’s accelerated gradient */
6. compute $\nabla f_\mu(z^{(k)})$;
7. set $\alpha_k = \frac{k+1}{2}$ and $\tau_k = \frac{2}{k+3}$;
8. /* solve: */
9. $\bar{q} = \arg\min_z \frac{L_\mu}{2} \|z - z^{(k)}\|_2^2 + \langle \nabla f_\mu(z^{(k)}), z - z^{(k)} \rangle$ s.t. $\|Az - y\|_\infty \leq \varepsilon$;
10. $\bar{w} = \arg\min_z \frac{L_\mu}{2} \|z - z^{(0)}\|_2^2 + \sum_{i=0}^{k} \alpha_i \langle \nabla f_\mu(z^{(i)}), z - z^{(k)} \rangle$ s.t. $\|Az - y\|_\infty \leq \varepsilon$;
11. /* update $z$ */
12. $z^{(k+1)} = \tau_k \bar{w} + (1 - \tau_k) \bar{q}$;
13. /* stopping criterion */
14. if $\|z^{(k+1)} - z^{(k)}\|_\infty < \tau$ then
15. $z^{(K)} = z^{(k+1)}$; break loop
16. end
17. /* decrease the value of $\mu$ */
18. set $\mu_t = \gamma \mu_{t-1}$;
19. set $z^{(0)} = z^{(K)}$;
20. end
21. return $z^{(K)}$.

Algorithm 2: NESTA for solving Problem 2.10
The pseudo-code of NESTA, tailored to (2.10), is given in Algorithm 2. The outer iterations in Line 3 iterate for decreasing values of \( \mu \), starting at an initial value \( \mu_0 \) (computed in Line 1) till a user-specified final value \( \mu_f \). The user also specifies the numbers of outer iterations \( T \), such that at each iteration the value of \( \mu \) is decreased by an amount \( \gamma < 1 \), computed in Line 2; the value of \( \mu \) is decreased after each outer iteration, as shown in Line 15. The choice of \( \mu_f \) implies a trade-off between the speed of convergence (the convergence rate of solving Equation 2.31 is proportional to the \( \mu \) used in each iteration) and the accuracy of the smoothed approximation \( f_\mu \), which consequently determines the NESTA’s overall accuracy. According to experiments in [138], decreasing \( \mu_f \) by a factor of 10 gives about 1 additional digit of accuracy on the optimal value.

NESTA uses a warm start mechanism, such that the solution \( z^{(K)} \) for a given \( \mu \) is used as initial guess at the next iteration, as shown in Line 16. Choosing a good initial guess for the first iteration (input \( z^{(0)} \) in Algorithm 2) may also contribute to speed-up the solver. In our tests we used the naive solution (linear interpolation) as initial guess for NESTA.

For a given value of \( \mu \), Line 5-Line 12 describe Nesterov’s accelerated gradient method applied to the smooth problem with objective \( f_\mu \). The accelerated gradient method involves \( K \) inner iterations (Line 5) and terminates if the change in the depth estimate is small (stopping condition in Line 11-Line 12). Nesterov’s method updates the depth estimate \( z^{(k+1)} \) (Line 10) using a linear combination of intermediate variables \( \bar{q} \) (Line 8) and \( \bar{w} \) (Line 9). We refer the reader to [160] for more details. We provide closed-form expressions for the gradient \( \nabla f_\mu (z) \) and for the vectors \( \bar{q} \) and \( \bar{w} \) (lines 8-9) in Section A.13.

Note that when \( \varepsilon = 0 \), Algorithm 2 solves the noiseless problem (2.9). This only affects the closed-form solutions for \( \bar{q} \) and \( \bar{w} \), but does not alter the overall structure of the algorithm. Similarly, Algorithm 2 can be used to solve problems (2.15) and (2.17), after replacing the matrix \( D \) with \( \Delta \) in the definition of \( f(z) \). As discussed earlier, the choice of a nonzero \( \mu_f \) in NESTA will result in an approximate solution to the optimal solution of (2.10). Consequently, NESTA may produce slightly less accurate solutions, while being much faster than \texttt{cvx}. Our experimental results show that the accuracy loss is negligible if the parameter \( \mu_f \) is chosen appropriately, see Section 2.6.2.

2.6 Experiments

This section validates our theoretical derivations with experiments on synthetic, simulated, and real data. Empirical evidence shows that our recovery techniques perform very well in practice, in both 2D and 3D environments. Our algorithm is also more robust to noise than a naive linear interpolation, and outperforms previous work in both reconstruction accuracy and computational speed.

We discuss a number of applications, including 2D mapping (Section 2.6.1), 3D depth reconstruction from sparse measurements (Section 2.6.3), and super-resolution depth imaging (Section 2.6.4). For the 3D case, we also provide a Monte Carlo analysis comparing the different solvers and choices of the objective functions (Section 2.6.2).

In the following tests, we evaluate the accuracy of the reconstruction by the average pixel-wise depth error, i.e., \( \frac{1}{n} \| z^* - z^0 \|_1 \), where \( z^0 \) is the ground truth and \( z^* \) is the reconstruction, unless otherwise specified.
2.6.1 2D Sparse Reconstruction and Mapping

In this section, we apply our algorithm to reconstruct 2D depth signals (e.g., the data returned by a 2D laser scanner). We provide both a statistical analysis on randomly generated synthetic signals (Sections 2.6.1-2.6.1), and a realistic example of application to 2D mapping (Section 2.6.1).

Typical Examples of 2D Reconstruction

![Figure 2-10](image)

Figure 2-10: An example of synthetic 2D signal and typical behavior of the compared techniques, naive, L1, and A1, for noise level $\varepsilon = 0.5m$.

We create a synthetic dataset that contains random piecewise linear depth signals of size $n = 2000$, with given number of corners. Since the number of variables is small, we use cvx/MOSEK [158, 159] as solver in all 2D experiments. When possible, we compare three different reconstruction algorithms: (i) the linear interpolation produced by Matlab’s command `interp1`, denoted as naive, (ii) the estimate from (2.9) (noiseless case) or (2.10) (noisy case), denoted as L1, and (iii) the estimate produced by Algorithm 1, denoted as A1.

An example of synthetic 2D signal (with only one corner) is shown in Figure 2-10. The green line is the ground truth signal, while the others are reconstructed depth profiles from sparse and noisy measurements using the three different algorithms.

Figure 2-10 provides a typical example of 2D reconstruction results. naive linearly interpolates the samples, hence even when measuring all depth data, it still produces a jagged line, due to measurement noise. It is easy to show that when measurement noise is uniformly distributed in $[-\varepsilon, +\varepsilon]$ (as in our tests), the average error committed by naive converges to $\varepsilon/2$ for increasing number of samples. In the figure, we consider $\varepsilon = 0.05m$. On the other hand, L1 and A1 correctly smooth the noise out. In particular, while L1 returns a (sign consistent) solution that typically has rounded corners, A1 is able to rectify these errors, producing an estimate that, even in the noisy case, is very close to the truth.
This section presents a Monte Carlo analysis of the reconstruction errors and timing, comparing naive, L1, and A1. Results are averaged over 50 runs, and the synthetic 2D signals are generated as specified in the previous section.

Figure 2-11(a) shows how the depth reconstruction quality is influenced by the number of corners in the ground truth signal (i.e., the sparsity of the true signal), comparing naive, L1, and A1. These results consider noiseless measurements and sample set including a twin sample in each linear region (these are the assumptions of Proposition 14). As predicted by Corollary 26, A1 recovers the original signal exactly (zero error). naive has large errors, while the L1 estimate falls between the two.

Figure 2-11(b) considers a more realistic setup: since in practice we do not know where
the corners are (hence we cannot guarantee to sample each linear segment of the true signal), in this case we uniformly sample depth measurements and we consider noisy measurements with $\varepsilon = 0.1m$. The figure reports the estimation error for increasing number of samples. As the percentage of samples goes to 1 (100%), we sample all entries of the depth signal. We consider signals with 3 corners in this test. The figure shows that for increasing number of samples, our approaches largely outperform the naive approach. $A_1$ improves over $L_1$ even in presence of noise, while the improvement is not as substantial as in the noiseless case of Figure 2-11(a). Figure 2-11(b) also shows that the error committed by naive does not improve when adding more samples. This can be understood from Figure 2-10 and the discussion in Section 2.6.1.

Figure 2-11(c) considers a fixed amount of samples (5%) and tests the three approaches for increasing measurement noise. Our techniques ($L_1, A_1$), are very resilient to noise and degrade gracefully in presence of large noise (e.g., $\varepsilon = 1m$).

Figure 2-11(d) shows the CPU times required by $L_1$ and $A_1$ in 2D reconstruction problems using the cvx solver. The CPU time for naive is negligible (in the milliseconds).

2D mapping from sparse measurements

This section applies our approach to a 2D mapping problem from sparse measurements. We use the Stage simulator [162] to simulate a robot equipped with a laser scanner with only 10 beams, moving in a 2D scenario. The robot is in charge of mapping the scenario; we assume the trajectory to be given. Our approach works as follows: we feed the 10 samples measured by our “sparse laser” to algorithm $A_1$; $A_1$ returns a full scan (covering 180 degrees with 180 scans in our tests), which we feed to a standard mapping routine (we use gmapping [163] in our tests).

Figure 2-12 compares the occupancy grid map produced by a standard mapping algorithm based on a conventional laser scan (Figure 2-12(a)), against the occupancy grid map reconstructed from our 10-beam laser. Figure 2-12(b) shows the map produced from the scans estimated using naive: the map has multiple artifacts. Figure 2-12(c) shows the...
map produced from the scans estimated using $A_1$; the proposed technique produces a fairly accurate reconstruction from very partial information.

## 2.6.2 3D Reconstruction: Datasets, Objective Functions and Solvers

This section introduces the 3D datasets used for the evaluation in the following sections. Moreover, it provides a statistical analysis of the performance obtained by the algorithmic variants presented in Section 2.5.1, as well as the solvers presented in Section 2.5.2. The best performing variants and solvers will be used in the real-world examples and applications presented in Sections 2.6.3-2.6.4.

### Datasets

In this section we introduce the datasets we use to benchmark our 3D depth reconstruction approaches. In order to have a ground truth signal, we collected several datasets with commonly-used high-resolution depth sensors (including a *Kinect* and a *ZED* stereo camera) and use an heavily down-sampled depth image as our “sparse” depth measurements. Moreover, we created synthetic signals for a more exhaustive evaluation.

![Figure 2-13: (a) Gazebo Simulated data. (b) ZED Stereo data. (c)-(d) Kinect data.](image)

Figure 2-13: (a) *Gazebo* Simulated data. (b) *ZED* Stereo data. (c)-(d) *Kinect* data.
Our testing datasets include a dataset of randomly-generated synthetic piecewise linear depth images (denoted as \( PL \)), a simulated dataset from the \textit{Gazebo} simulator [164] (denoted as \textit{Gazebo}), a stereo dataset from a \textit{ZED} camera (denoted as \textit{ZED}), 8 datasets from a \textit{Kinect} camera (denoted as \textit{K1} to \textit{K8}), the Middlebury stereo datasets [165, 166], the NYU Depth Dataset [30], and the KITTI Dataset [167]. More specifically, \textit{Gazebo} contains 20 full depth and RGB images rendered in an office-like environment from the \textit{Gazebo} simulator (Figure 2-13(a)). \textit{ZED} includes 1000 full disparity and RGB images, collected from a \textit{ZED} stereo camera mounted on a dolly, in the \textit{Laboratory of Information and Decision Systems (LIDS)} at MIT (Figure 2-13(b)). \textit{K1} to \textit{K8} contain odometry information, as well as depth and RGB images, collected from a \textit{Kinect} sensor mounted on a dolly with wheel odometers, moving in 8 different locations at MIT, including tunnels, offices, and corridors (Figure 2-13(c)-(d)). The Middlebury stereo dataset is used for the sake of benchmarking against the previous works [4, 5], which use a similar experimental setup, and includes disparity images of size 256-by-256 (each down-sampled from the original 512-by-512 images). Both the NYU and the KITTI datasets are used for comparison against deep-learning based approaches.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure.png}
\caption{Comparison between three different objective functions, \textit{L1}, \textit{L1diag}, and \textit{L1cart}, on 10 benchmarking datasets. (a) constant in \textit{L1cart} is chosen as \( \delta = 0.1 \text{m} \), (b) constant in \textit{L1cart} is chosen as \( \delta = 0.01 \text{m} \).}
\end{figure}

\section*{Objective Functions}

In this section we compare the three objective functions discussed in Section 2.5.1 for the noiseless reconstruction problem 2.15. We use the \texttt{cvx/MOSEK} [158] solver in MATLAB in this section, to reduce numerical approximations, while we evaluate the use of other solvers (NESTA) in the next section.

Figure 2-14 compares the reconstruction errors of the three different objective functions on the datasets \textit{PL}, \textit{ZED}, and \textit{K1-K8}. The error bars show the reconstruction error for each objective functions (\textit{L1}, \textit{L1diag}, and \textit{L1cart}), averaged over all the images in the corresponding dataset. The depth measurements are sampled from a grid, such that only 4%
of the pixels in the depth signals are used. The ground truth signals have resolution $85 \times 103$ for the Kinect datasets, $96 \times 128$ for the ZED dataset, and $40 \times 40$ for the PL dataset.

From Section 2.5.1, we recall that $L_1\text{cart}$ includes a parameter $\delta$ which prevents the denominator of some of the entries in (2.29) to become zero. Figure 2-14(a) and Figure 2-14(b) show the reconstruction errors for $\delta = 0.1m$ and $\delta = 0.01m$, respectively. From Figure 2-14 it is clear that the accuracy of $L_1\text{cart}$ heavily depends on the choice of $\delta$, and degrades significantly for small values of $\delta$. Moreover, even for a good choice of $\delta$ (Figure 2-14(a)) the advantage of $L_1\text{cart}$ over $L_1$ and $L_1\text{diag}$ is minor in most datasets. The $L_1\text{diag}$ objective, on the other hand, performs consistently better than $L_1$ across all datasets and is parameter-free.

We conclude that while the variants $L_1$, $L_1\text{diag}$, and $L_1\text{cart}$ do not induce large performance variations, $L_1\text{diag}$ ensure accurate depth reconstruction and we focus our attention on this technique in the following sections.

Solvers

This section compares two solvers for $\ell_1$-minimization in terms of accuracy and speed. The first solver is CVX/MOSEK [158] (denoted as CVX for simplicity), a popular general-purpose parser/solver for convex optimization. The second is NESTA [138], which we adapted to our problem setup in Section 2.5.2. We implemented NESTA in Matlab, starting from the open-source implementation of [138]; our source code is also available at https://github.com/sparse-depth-sensing.

We compare the two solvers on the synthetic dataset PL, using the $L_1\text{diag}$ objective function. Each depth image in PL is generated randomly with a fixed number of corners (3 in our tests) and is of size 100-by-100, unless otherwise specified. All depth measurements are uniformly sampled at random from the ground truth signal, and the 4 immediate neighbors (up, down, left, right) are also added into the sample sets. No noise is injected into the measurements ($\varepsilon = 0$). In all tests, we set the maximum number of inner iterations to $K = 10000$, the number of continuation steps to $T = 5$, and the stopping criterion to $\tau = 10^{-5}$ for NESTA. All data points in the plots are averaged from 50 random runs.

We start by evaluating the impact of the parameter $\mu_f$ on the accuracy and timing of NESTA. Figure 2-15 shows the trade-off between reconstruction error and computational time for different values of $\mu_f$. The error is evaluated as the average mismatch between NESTA and CVX solutions. In each test, the depth samples include 5% of the pixels, uniformly chosen at random. We note that the average error is in the order of millimeters in all cases. To obtain the best trade-off between accuracy and speed, we choose $\mu_f = 0.001$, the “elbow” point in Figure 2-15. We use this value in all the following experiments.

Figure 2-16 compares the performance of CVX and NESTA for increasing number of samples, noise, and size of the depth signals. Figure 2-16(a)-(b) show the reconstruction error and computational time of the two solvers for increasing percentage of samples. Depth measurements are affected by entry-wise uniformly random measurement noise in $[-\varepsilon, \varepsilon]$; for this test we chose $\varepsilon = 0.1$. Figure 2-16(a) shows that the accuracy of NESTA is close to the one of CVX (the mismatch is in the order of few millimeters), while they both largely outperform Matlab’s linear interpolation (naive). Figure 2-16(b) shows that NESTA is around 10x faster than CVX (as in the 2D case, the computational time of naive is negligible).
Figure 2-15: Trade-off between accuracy and speed for the NESTA solver with different parameter values $\mu_f$. As $\mu_f$ decreases, NESTA produces a more accurate solution at the cost of higher computational time. The error is computed as the average mismatch between the NESTA and cvx solutions.

Figure 2-16(c)-(d) show the reconstruction error and computational time for increasing noise level when sampling 5% of the depth signal. Also in this case the errors of NESTA and cvx are very close, while NESTA remains remarkably faster. For both NESTA and cvx, the estimation error grows more gracefully with respect to the measurement noise $\varepsilon$, compared to the naive approach.

Figure 2-16(e)-(f) show the reconstruction error and computational time for increasing image size. We reconstruct random signals of size $N$-by-$N$ using 5% of samples, without adding noise. Figure 2-16(e) further confirms that the error curves for cvx and NESTA are almost indistinguishable, implying that they produce reconstructions of similar quality. However, the NESTA solver entails a speed up of 3-10x, depending on the problem instance (Figure 2-16(f)).

Given the significant advantage of NESTA over cvx, and since cvx is not able to scale to large signals, we use NESTA in the tests presented in the following sections.

### 2.6.3 Single-Frame Sparse 3D Reconstruction

The previous section confirmed that choosing $L1\text{diag}$ as objective function and NESTA (with $\mu_f = 0.001$) as solver ensure the best performance. This section extends the numerical evaluation to the other 3D datasets, including Gazebo, ZED, K1-K8. For each dataset, we use $L1\text{diag}$ to reconstruct the depth at each frame from a small subset of samples, and we compare our approach against the naive linear interpolation. In the following,
Figure 2-16: Comparison between NESTA ($\mu_f = 0.001$) and CVX. Estimation errors and timing are shown for (a-b) increasing number of samples, (c-d) increasing measurement noise, (e-f) increasing size of the $N \times N$ depth signals. NESTA achieves comparable reconstruction errors, while offering a significant speedup.
we discuss typical reconstruction results, provide error statistics for different percentages of samples and noise levels, and compare $L_{\text{diag}}$ against the state-of-the-art techniques proposed in [4, 5].

**Typical Examples of 3D Reconstruction**

We start by showing reconstruction examples from sparse depth measurements on the Gazebo and K1 datasets.

![Figure 2-17](image)

Figure 2-17: The first row is an example of sparse depth reconstruction on Gazebo simulated data: (a) RGB image, (b) uniformly drawn sparse samples, and (c) reconstruction using $L_{\text{diag}}$. The second row is an example on Kinect K1 data: (d) RGB image, (e) sparse samples on a grid, and (f) reconstruction using $L_{\text{diag}}$.

Figure 2-17(a)-(c) show an example on the Gazebo simulated dataset with uniformly random depth measurements and the reconstructed full depth signal based on these samples. The reconstructed depth image reflects the true geometry of the scene, even when we are only using 2% samples and their neighbors (total is roughly 8%). The reconstruction error in this example is 5cm.

Figure 2-17 (d)-(e) shows an example on the Kinect K1 dataset, where all depth measurements fall on a regular grid. This sampling strategy resembles the output of a low-resolution depth sensor. Note that even though only a total number of 42 measurements is available, the reconstructed depth image still correctly identifies the corridor and the walls. The reconstruction error in this example is 18cm.

Extra visualizations for the Gazebo and the ZED datasets are provided in Figure 2-18 and Figure 2-19, respectively.
Figure 2-18: Qualitative results on Gazebo dataset: 3 examples of reconstructed depth signals using the proposed approaches ($L_1$, $L_1\text{diag}$) and a naive linear interpolation (naive). For each example we show the reconstruction from 2% uniformly drawn depth measurements. We also show the reconstruction for the case in which we can access the depth corresponding to (appearance) edges in the RGB images.
Figure 2-19: Qualitative results on ZED: 3 examples of reconstructed depth signals using the proposed approaches ($L_1$, $L_{1diag}$) and a naive linear interpolation ($\text{naive}$). For each example we show the reconstruction from 2% uniformly drawn depth measurements. We also show the reconstruction for the case in which we can access the depth corresponding to (appearance) edges in the RGB images.
Statistics for 3D Reconstruction

In this section we rigorously benchmark the performance of L1diag against the naive approach, in terms of both the reconstruction accuracy and the robustness to measurement noise.

Figure 2-20 depicts the reconstruction errors for increasing percentages of uniformly-random samples on different datasets. Figure 2-20(a) shows reconstruction from noiseless samples on the Gazebo simulated datasets, while Figure 2-20(b) is the same plot except with additional pixel-wise independent Gaussian measurement noise $\varepsilon = 0.1$. Figure 2-20(c)-(d) show the experimental results on the ZED stereo dataset and K1 dataset. No additional noise is added to these two datasets, since the raw data is already affected by actual sensor noise. Figure 2-20(e) shows the comparison between naive and L1diag over all datasets for reconstructions from 10% samples and their immediate neighbors.

From the figures it is clear that our approach consistently outperforms the naive linear interpolation in both the noiseless and noisy settings and across different datasets. The gap between L1diag and naive widens as the number of samples increases in the noisy setup, which demonstrates that our approach is more resilient to noise. In the noiseless setup, the gap shrinks as the percentage of samples converges to 100%, since in this case we are sampling a large portion of the depth signal, a regime in which the naive interpolation often provides a satisfactory approximation. L1diag produces significantly more accurate reconstruction (20-50% error reduction compared with naive) when operating below the 20%-samples regime, which is the sparse sensing setup that motivated this work in the first place.

Comparisons on Middlebury Dataset

In this section we provide an empirical comparison of our algorithm against a number of prior work on disparity image reconstruction from sparse measurements. Following the experimental setup used in [4, 5], we benchmark our technique in the Middlebury\(^4\) stereo datasets [165, 166]. Six different disparity images of size 256-by-256 (each downsampled from the original 512-by-512 images) are selected from the Middlebury dataset. We evaluate both the reconstruction accuracy and computational times for 4 different algorithms, including naive and L1diag (discussed earlier in this chapter), Hawe’s CSR [4], and Liu’s WT+CT [5]. The sparse measurements are uniformly sampled from the ground truth image without noise. The same set of sparse samples are used for all 4 methods in each set of experiments. In order to allow a closer comparison with [4, 5] in this section we use the peak signal-to-noise ratio (PSNR) as a measure of reconstruction accuracy, where a higher PSNR indicates a better reconstruction. The PSNR is defined as follows, where $z$ is the reconstruction, $z^\circ$ is the ground truth, and $n$ is the dimension of the vectorized signal:

$$\text{PSNR} = 20 \cdot \log_{10} \max z^\circ - 10 \cdot \log_{10} \left( \frac{1}{n} \sum_{i=1}^{n} [z_i - z^\circ_i]^2 \right)$$

\(^4\)http://vision.middlebury.edu/stereo/data/
Figure 2-20: Reconstruction errors for increasing percentage of uniform samples, and for different datasets. (a) and (b) are reconstructions on the Gazebo dataset, using noiseless and noisy (noise bounded by $\epsilon = 0.1m$) samples, respectively. (c) reconstruction on the ZED dataset, (d) reconstruction on the K1 dataset. (e) comparison on all datasets.
To ensure a fair comparison, the initial setup (e.g., memory allocation for matrices, building a constant wavelet/contourlet dictionary) has been excluded from timing. All algorithms are initiated without warm-start, meaning that the sample image (rather than the result from naive) is used as the initial guess to our optimization problems. For L1diag, we use NESTA as solver with the same settings specified in Section 2.6.2. For WT+CT, we set 100 as the maximum number of iterations, which strikes the best trade-off between accuracy and timing.

Table 2.2 reports the results of our evaluation, for each image in the Middlebury dataset (rows in the table), and for increasing number of samples (columns in the table). For each cell, we report the PSNR in dB and the time in seconds. The runtime is measured on Intel Xeon(R) CPU E5-2650 v3, with 20 cores at 2.30GHz. However, all methods run on a single-thread. A cell is marked as N/A if the PSNR falls below 20dB [4, 5], which indicates that either the algorithm fails to converge or that the reconstructed image is significantly different than the ground truth. Best accuracy and best timing are highlighted in bold (recall that the higher the PSNR the better).

Our proposed algorithm L1diag consistently outperforms all other algorithms in terms of accuracy in every single experiments. In addition, L1diag is the only algorithm that ensures acceptable performance at aggressively low sampling rates (as low as 0.5%), while both [4] and [5] fail with 1% samples or fewer. L1diag is significantly faster than both [4] and [5]. For instance, L1diag takes only 50% to 10% of the computational time of WT+CT, depending on the number of samples. The naive interpolation is very fast, but produces worse reconstruction than L1diag. We noticed that in these tests we can achieve even faster runtime for L1diag by using a larger parameter $\mu_f$ without suffering much loss in accuracy. For instance, for $\mu_f = 0.1$, the average computation time with 5% samples

---

Table 2.2
<table>
<thead>
<tr>
<th>Image</th>
<th>5% samples</th>
<th>CSR</th>
<th>WT+CT</th>
<th>naive</th>
<th>L1diag</th>
<th>ground truth</th>
<th>rgb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aloe</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baby</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Art</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2-21: Examples of reconstruction from 5% uniformly random samples on the Middlebury disparity dataset, using 4 different algorithms: naive, L1diag, CSR [4], and WT+CT [5]. The proposed algorithm, L1diag, is able to preserve sharper boundaries and finer details, while not creating jagged edges as in naive.
Figure 2-22: Reconstructed depth profile of the “Art” image from the Middlebury dataset from 5% uniform random measurements, visualized as point cloud. Red indicates errors and green highlights good reconstruction quality. In comparison against naive and WT+CT [5], L1diag produces the most complete structures with the least outliers.
<table>
<thead>
<tr>
<th>Name</th>
<th>Method</th>
<th>PSNR (dB) / Time (s) (Percentage of Samples)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.5% 1% 5% 10%</td>
</tr>
<tr>
<td>Aloe</td>
<td>CSR</td>
<td>N/A  N/A 21.4 / 10.6 24.1 / 8.65</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  N/A 21.9 / 19.4 24.3 / 19.5</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>N/A  21.6 / 0.17 24.7 / 0.14 26.0 / 0.22</td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>20.6 / 14.5</strong> <strong>21.7 / 7.02</strong> <strong>24.9 / 3.12</strong> <strong>26.4 / 2.06</strong></td>
</tr>
<tr>
<td>Art</td>
<td>CSR</td>
<td>N/A  N/A 23.1 / 11.9 25.3 / 9.80</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  N/A 25.0 / 19.8 26.7 / 19.5</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>21.9 / <strong>0.15</strong> 23.5 / <strong>0.16</strong> 26.3 / <strong>0.17</strong> 27.7 / <strong>0.18</strong></td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>22.5 / 11.1</strong> <strong>23.8 / 8.86</strong> <strong>26.6 / 3.78</strong> <strong>27.8 / 2.23</strong></td>
</tr>
<tr>
<td>Baby</td>
<td>CSR</td>
<td>N/A  N/A 26.6 / 10.0 31.1 / 9.11</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  24.1 / 19.6 27.7 / 19.4 31.5 / 19.5</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>27.6 / <strong>0.15</strong> 27.4 / <strong>0.16</strong> 31.3 / <strong>0.16</strong> 33.3 / <strong>0.18</strong></td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>27.8 / 12.1</strong> <strong>28.4 / 10.5</strong> <strong>32.5 / 3.21</strong> <strong>33.9 / 2.06</strong></td>
</tr>
<tr>
<td>Dolls</td>
<td>CSR</td>
<td>N/A  N/A 24.3 / 13.2 26.5 / 11.0</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  20.6 / 19.5 27.5 / 19.6 28.2 / 20.3</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>25.8 / <strong>0.13</strong> 24.5 / <strong>0.16</strong> 27.8 / <strong>0.16</strong> 28.5 / <strong>0.18</strong></td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>26.9 / 7.07</strong> <strong>27.5 / 5.49</strong> <strong>28.3 / 2.24</strong> <strong>28.9 / 3.03</strong></td>
</tr>
<tr>
<td>Moebius</td>
<td>CSR</td>
<td>N/A  N/A 23.6 / 11.9 26.1 / 10.5</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  22.4 / 19.3 26.3 / 19.5 27.6 / 19.4</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>25.7 / <strong>0.14</strong> 24.7 / <strong>0.16</strong> 26.8 / <strong>0.15</strong> 27.8 / <strong>0.18</strong></td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>25.8 / 6.91</strong> <strong>26.4 / 7.03</strong> <strong>27.5 / 2.90</strong> <strong>28.6 / 2.59</strong></td>
</tr>
<tr>
<td>Rocks</td>
<td>CSR</td>
<td>N/A  N/A 23.1 / 11.5 25.0 / 9.15</td>
</tr>
<tr>
<td></td>
<td>WT+CT</td>
<td>N/A  23.2 / 19.3 25.6 / 19.2</td>
</tr>
<tr>
<td></td>
<td>naive</td>
<td>21.7 / <strong>0.15</strong> 23.8 / <strong>0.15</strong> 25.8 / <strong>0.15</strong> 27.2 / <strong>0.19</strong></td>
</tr>
<tr>
<td></td>
<td>L1diag</td>
<td><strong>22.7 / 12.0</strong> <strong>24.3 / 9.71</strong> <strong>25.9 / 3.22</strong> <strong>27.3 / 2.68</strong></td>
</tr>
</tbody>
</table>

Table 2.2: Reconstruction accuracy and computational time comparing naive, L1diag, CSR [4], and WT+CT [5]. L1diag consistently outperforms all other methods in accuracy, and performs robustly even with aggressively low number of measurements.
reduces from around 3s to around 2s, while the PSNR remains at roughly the same level and still outperforms other approaches.

For a visual comparison of the reconstructed depth images, Figure 2-21 reports some examples of the reconstructed disparity images for each of the compared techniques. The proposed algorithm, L1diag, is able to preserve sharp boundaries and fine details, while avoiding the creation of jagged edges as in naive.

We also demonstrate examples of reconstructed depth profile as point cloud in Figure 2-22. naive creates a large number of outliers due to extrapolation, and mistakenly forms lines between spatially disconnected points. WT+CT [5] results in some distortion and broken structures, especially in the eraser at the bottom as well as the ring at the top right. In comparison, L1diag produces the most complete structures with the least outliers.

**Comparisons on NYU Dataset**

In this section, we compare our proposed algorithm against a state-of-the-art, deep-learning-based method [7] for depth completion, on the NYU Depth V2 dataset [30]. Specifically, [7] use a deep neural network with an encoder-decoder structure. The network is trained for depth prediction with both RGB and sparse depth as input, and a dense depth image as output. In comparison, our proposed method does not rely on any parameter tuning, and is tested on the official test dataset directly.

We adopt the standard error metrics used in depth prediction tasks, including RMS (root mean squared error), MAE (mean absolute error), and $\delta_1$ (percentage of predicted pixels with a relative error below 25%, higher is better). The experimental results with 1% uniform random samples (roughly 693 samples per image) are listed in Table 2.3.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMS[cm]</th>
<th>MAE[cm]</th>
<th>$\delta_1$[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>20.4</td>
<td>12.1</td>
<td>97.9</td>
</tr>
<tr>
<td>naive</td>
<td>17.7</td>
<td>7.58</td>
<td>98.3</td>
</tr>
<tr>
<td>L1diag</td>
<td>15.8</td>
<td>6.57</td>
<td>98.7</td>
</tr>
</tbody>
</table>

Table 2.3: Our proposed solution outperforms state-of-the-art deep-learning based method [7] on the NYU test dataset. The input depth is sampled uniform randomly (1%) from the ground truth.

From Table 2.3, it is clear that on the NYU dataset, our proposed solution outperforms both naive and the competitive deep-learning methods. More surprisingly, even naive produces higher-accuracy than [7]. This observation can be attributed to the two facts: (1) the depth images in the NYU dataset align well with our assumptions regarding well-structured indoor environments; (2) the uniform sampling strategy guarantees dispersed samples in the pixel space, and thus the inverse problem is sufficiently regulated.

**Comparisons on KITTI Dataset**

In contrast, on the outdoor dataset KITTI [121], our assumption on well-structured environments does not hold due to complex objects like trees, pedestrians, and cyclists. In addition,
the lidar measurements are available mostly at the bottom half of the image, rather than uniformly distributed over the entire image space. Consequently, both $L_{1\text{diag}}$ and naive fail to produce meaningful reconstructions, but [7] is still able to generate predictions with high accuracy.

![Image](a) 72-by-103 depth image  ![Image](b) up-scaled 359-by-512 image

Figure 2-23: Super-resolution depth imaging. The up-scale factor is 24.79 in this example.

### 2.6.4 Super-dense 3D Reconstruction and Super-resolution Depth Imaging

In this section, we demonstrate that our algorithm can also be applied to super-resolution depth imaging. Super-resolution imaging attempts to algorithmically increase the resolution of a given depth signal. This is fundamentally the same as viewing an input full depth signal as measurements sampled from a higher-resolution “ground truth” and do reconstruction based on such measurements. An example is shown in Figure 2-23 using Kinect data. The original signal, in Figure 2-23(a), has a resolution of 72-by-103 and many missing pixels (due to Kinect sensor noise). Figure 2-23(b) shows the reconstructed, or in other words up-scaled, depth image. The size of the reconstructed depth image is 359-by-512, and thus has an up-scale factor of 24.79. Roughly speaking, 1 depth pixel in the input signal translates to a 5-by-5 patch in the up-scaled depth signal. Note that all missing pixels (including the legs of the chair in Figure 2-23(a)) are smoothed out.

### 2.7 Summary

In this chapter, we propose a new approach to recover dense 2D and 3D depth signals from sparse and incomplete depth measurements. As a first contribution, we formulate depth reconstruction as the problem of finding a signal that has the sparsest second-order derivative, i.e., the least amount of corners and edges, while matching the given measurements. The problem itself is NP-hard, hence we relax it to a convex $\ell_1$-minimization problem with $\ell_\infty$-norm constraints.
Our second contribution is a theoretical analysis that establishes precise conditions under which the dense depth signal can be recovered from sparse samples. Even in the case in which exact recovery is not possible, we provide error bounds on the estimated signal and discuss its sensitivity to measurement noise in both 2D and 3D problems.

As a third contribution, we present several algorithmic variants to recover the depth signal, each one resulting in a convex optimization problem. To further accelerate these algorithms, we discuss how to adapt NESTA, a first-order method for nonsmooth optimization, to our problem setup.

The fourth contribution is an extensive experimental evaluation on both synthetic and real data. The experimental results show that our algorithms are able to reconstruct a dense depth signal from an extremely low number of measurements (e.g., we can recover a 100-by-100 depth signal from 40 measurements), are robust to measurement noise, and are able to scale to large signals. The capability of properly modeling measurement noise enables a performance boost with respect to interpolation-based approaches. We demonstrated the proposed approach in many applications, including 2D mapping, single-frame and multi-frame 3D depth reconstruction from sparse measurements, 3D depth signal compression and decompression, as well as super-resolution depth imaging.

As future work, we plan to further accelerate our algorithms using parallel computing (e.g., GPU). We would also like to apply the proposed algorithm to distributed mapping in bandwidth-limited multi-robot systems. In addition, we are interested in developing motion planning algorithms that can pro-actively guide the depth sampling process and further improve the reconstruction results.
Chapter 3

Deep Regression Network and Self-Supervised Learning

In this chapter, we present our second contribution: a deep regression network for depth completion, as well as a self-supervised learning framework to train the network without using ground truth dense annotations. Compared to the model-based approach in Chapter 2, the second contribution in this chapter is pure data-driven and the quality of reconstruction relies heavily on the training dataset. In particular, we consider the depth completion problem with both sparse depth and color image as input. We use a deep regression model that takes both inputs and predicts a full-resolution depth image.

This chapter consists of three main parts. First, we focus on supervised training (i.e., when we do have access to ground truth depth annotations) in Section 3.1 and its results Section 3.2. Second, in Section 3.3, we demonstrate the application of depth completion in sparse map densification, as is mentioned in Section 1.3.3. Third, we describe our proposed self-supervised learning framework in Section 3.4 and Section 3.5, to address the problem that dense annotations for depth are difficult to acquire in practice.

3.1 Supervised Training

In this section, we describe the architecture of the convolutional neural network. We also discuss the depth sampling strategy, the data augmentation techniques, and the loss functions used for training.

3.1.1 CNN Architecture

We found in our experiments that many bottleneck architectures (with an encoder and a decoder) could result in good performance. We chose the final structure based on [111] for the sake of benchmarking, because it achieved state-of-the-art accuracy in RGB-based depth prediction. The final structure is illustrated in Figure 3-1.

The feature extraction (encoding) layers of the network, highlighted in blue, consist of a ResNet [6] followed by a convolution layer. More specifically, the ResNet-18 is used for

---

1 The work presented in this chapter has been published [88, 168, 169]
Figure 3-1: CNN architecture for NYU-Depth-v2 and KITTI datasets, respectively. The network for KITTI has fewer layers and channels, since KITTI has a larger image size than NYU-Depth-v2 and thus demands higher memory consumption. Cubes are feature maps, with dimensions represented as #features@height×width. The encoding layers in blue consist of a ResNet [6] and a 3×3 convolution. The decoding layers in yellow are composed of 4 upsampling layers (UpProj) followed by a bilinear upsampling.

KITTI, and ResNet-50 is used for NYU-Depth-v2. The last average pooling layer and linear transformation layer of the original ResNet have been removed. The second component of the encoding structure, the convolution layer, has a kernel size of 3-by-3.

The decoding layers, highlighted in yellow, are composed of 4 upsampling layers followed by a bilinear upsampling layer. We use the UpProj module proposed by Laina et al. [111] as our upsampling layer, but a deconvolution with larger kernel size can also achieve the same level of accuracy. An empirical comparison of different upsampling layers is shown in Section 3.2.1. Note that the number of feature maps for the KITTI network is only 25% of the counterpart in the NYU network.

### 3.1.2 Depth Sampling

During training, the input sparse depth $D$ is sampled randomly from the ground truth depth image $D^*$ on the fly. In particular, for any targeted number of depth samples $m$ (fixed during training), we compute a Bernoulli probability $p = \frac{m}{n}$, where $n$ is the total number of valid depth pixels in $D^*$. Then, for any pixel $(i, j)$,

$$D(i, j) = \begin{cases} D^*(i, j), & \text{with probability } p \\ 0, & \text{otherwise} \end{cases}$$

With this sampling strategy, the actual number of non-zero depth pixels varies for each training sample around the expectation $m$. Note that this sampling strategy is different from dropout [170], which scales up the output by $1/p$ during training to compensate for deactivated neurons. The purpose of our sampling strategy is to increase robustness of the network against different number of inputs and to create more training data (i.e., a data augmentation technique). It is worth exploring how injection of random noise and a different sampling strategy (e.g., feature points) would affect the performance of the network.
3.1.3 Data Augmentation

We augment the training data in an online manner with random transformations, including

- **Scale**: color images are scaled by a random number \( s \in [1, 1.5] \), and depths are divided by \( s \).
- **Rotation**: color and depths are both rotated with a random degree \( r \in [-5, 5] \).
- **Color Jitter**: the brightness, contrast, and saturation of color images are each scaled by \( k_i \in [0.6, 1.4] \).
- **Color Normalization**: RGB is normalized through mean subtraction and division by standard deviation.
- **Flips**: color and depths are both horizontally flipped with a 50% chance.

Nearest neighbor interpolation, rather than the more common bi-linear or bi-cubic interpolation, is used in both scaling and rotation to avoid creating spurious sparse depth points. We take the center crop from the augmented image so that the input size to the network is consistent.

3.1.4 Loss Function

One common and default choice of loss function for regression problems is the mean squared error (\( \mathcal{L}_2 \)). \( \mathcal{L}_2 \) is sensitive to outliers in the training data since it penalizes more heavily on larger errors. During our experiments we found that the \( \mathcal{L}_2 \) loss function also yields visually undesirable, over-smooth boundaries instead of sharp transitions.

Another common choice is the Reversed Huber (denoted as berHu) loss function [171], defined as

\[
B(e) = \begin{cases} 
|e|, & \text{if } |e| \leq c \\
\frac{e^2 + c^2}{2c}, & \text{otherwise}
\end{cases} 
\]

[111] uses a batch-dependent parameter \( c \), computed as 20% of the maximum absolute error over all pixels in a batch. Intuitively, berHu acts as the mean absolute error (\( \mathcal{L}_1 \)) when the element-wise error falls below \( c \), and behaves approximately as \( \mathcal{L}_2 \) when the error exceeds \( c \).

In our experiments, besides the aforementioned two loss functions, we also tested \( \mathcal{L}_1 \) and found that it produced slightly better results on the RGB-based depth prediction problem. As a result, we use \( \mathcal{L}_1 \) as our default choice throughout the chapter for its simplicity and performance.

3.1.5 Error Metrics

We evaluate each method using the following metrics:

- **rmse**: root mean squared error
• REL: mean absolute relative error

• \( \delta_i \): percentage of predicted pixels where the relative error is within a threshold. Specifically,

\[
\delta_i = \frac{\text{card}\left(\{\hat{y}_i : \max\{\hat{y}_i / y_i, y_i / \hat{y}_i\} < 1.25^i\}\right)}{\text{card}\left(\{y_i\}\right)},
\]

where \( y_i \) and \( \hat{y}_i \) are respectively the ground truth and the prediction, and \( \text{card} \) is the cardinality of a set. A higher \( \delta_i \) indicates better prediction.

### 3.2 Supervised Results

In this section we present all experimental results for supervised training. First, we evaluate the performance of our proposed method with different loss functions and network components on the prediction accuracy. Second, we compare the proposed method with state-of-the-art methods on both the NYU-Depth-v2 and the KITTI odometry datasets. Third, we explore the impact of number of sparse depth samples on the performance.

#### 3.2.1 Architecture Evaluation

In this section we present an empirical study on the impact of different loss functions and network components on the depth prediction accuracy. The results are listed in Table 3.1.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Loss</th>
<th>Encoder</th>
<th>Decoder</th>
<th>rmse</th>
<th>REL</th>
<th>( \delta_1 )</th>
<th>( \delta_2 )</th>
<th>( \delta_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB</td>
<td>( \mathcal{L}_2 )</td>
<td>Conv</td>
<td>DeConv (_2)</td>
<td>0.610</td>
<td>0.185</td>
<td>71.8</td>
<td>93.4</td>
<td>98.3</td>
</tr>
<tr>
<td>berHu</td>
<td>Conv</td>
<td>DeConv (_2)</td>
<td>0.554</td>
<td>0.163</td>
<td>77.5</td>
<td>94.8</td>
<td>98.7</td>
<td></td>
</tr>
<tr>
<td>( \mathcal{L}_1 )</td>
<td>Conv</td>
<td>DeConv (_2)</td>
<td>0.552</td>
<td>0.159</td>
<td>77.5</td>
<td>95.0</td>
<td>98.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conv</td>
<td>DeConv (_3)</td>
<td>0.533</td>
<td>0.151</td>
<td>79.0</td>
<td>95.4</td>
<td>98.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conv</td>
<td>UpConv</td>
<td>0.529</td>
<td>0.149</td>
<td>79.4</td>
<td>95.5</td>
<td>98.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conv</td>
<td>UpProj</td>
<td>0.528</td>
<td>0.144</td>
<td>80.3</td>
<td>95.2</td>
<td>98.7</td>
<td></td>
</tr>
<tr>
<td>RGBd</td>
<td>( \mathcal{L}_1 )</td>
<td>ChanDrop</td>
<td>UpProj</td>
<td>0.361</td>
<td>0.105</td>
<td>90.8</td>
<td>98.4</td>
<td>99.6</td>
</tr>
<tr>
<td></td>
<td>DepthWise</td>
<td>UpProj</td>
<td>0.261</td>
<td>0.054</td>
<td>96.2</td>
<td>99.2</td>
<td>99.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conv</td>
<td>UpProj</td>
<td>0.264</td>
<td>0.053</td>
<td>96.1</td>
<td>99.2</td>
<td>99.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Evaluation of loss functions, upsampling layers and the first convolution layer. RGBd has an average sparse depth input of 100 samples. (a) comparison of loss functions is listed in Row 1 - 3; (b) comparison of upsampling layers is in Row 2 - 4; (c) comparison of the first convolution layers is in the 3 bottom rows.

To compare the loss functions we use the same network architecture, where the upsampling layers are simple deconvolution with a 2×2 kernel (denoted as DeConv\(_2\)\(_2\)). \( \mathcal{L}_2 \), berHu and \( \mathcal{L}_1 \) loss functions are listed in the first three rows in Table 3.1 for comparison. As shown in the table, both berHu and \( \mathcal{L}_1 \) significantly outperform \( \mathcal{L}_2 \). In addition, \( \mathcal{L}_1 \) produces slightly better results than berHu. Therefore, we use \( \mathcal{L}_1 \) as our default choice of loss function.
We perform an empirical evaluation of different upsampling layers, including deconvolution with kernels of different sizes (DeConv$_2$ and DeConv$_3$), as well as the UpConv and UpProj modules proposed by Laina et al. [111]. The results are listed from row 3 to 6 in Table 3.1.

We make several observations. Firstly, deconvolution with a $3 \times 3$ kernel (i.e., DeConv$_3$) outperforms the same component with only a $2 \times 2$ kernel (i.e., DeConv$_2$) in every single metric. Secondly, since both DeConv$_3$ and UpConv have a receptive field of $3 \times 3$ (meaning each output neuron is computed from a neighborhood of 9 input neurons), they have comparable performance. Thirdly, with an even larger receptive field of $4 \times 4$, the UpProj module outperforms the others. We choose to use UpProj as a default choice.

Since our RGBd input data comes from different sensing modalities, its 4 input channels (R, G, B, and depth) have vastly different distributions and support. We perform a simple analysis on the first convolution layer and explore three different options.

The first option is the regular spatial convolution (Conv). The second option is depth-wise separable convolution (denoted as DepthWise), which consists of a spatial convolution performed independently on each input channel, followed by a pointwise convolution across different channels with a window size of 1. The third choice is channel dropout (denoted as ChanDrop), through which each input channel is preserved as is with some probability $p$, and zeroed out with probability $1 - p$.

The bottom 3 rows compare the results from the 3 options. The networks are trained using RGBd input with an average of 100 sparse input samples. DepthWise and Conv yield very similar results, and both significantly outperform the ChanDrop layer. Since the difference is small, for the sake of comparison consistency, we will use the convolution layer for all experiments.

### 3.2.2 Comparison with the State-of-the-Art

In this section, we compare with existing methods.

We compare with RGB-based approaches [8, 111, 172], as well as the fusion approach [96] that utilizes an additional 2D laser scanner mounted on a ground robot. The quantitative results are listed in Table 3.2.

Our first observation from Row 2 and Row 3 is that, with the same network architecture, we can achieve a slightly better result (albeit higher REL) by replacing the berHu loss function proposed in [111] with a simple $L_1$. Secondly, by comparing problem group RGB (Row 3) and problem group d (e.g., Row 4), we draw the conclusion that an extremely small set of 20 sparse depth samples (without color information) already produces significantly better predictions than using RGB. Thirdly, by comparing problem group d and problem group RGBd row by row with the same number of samples, it is clear that the color information does help improve the prediction accuracy. In other words, our proposed method is able to learn a suitable representation from both the RGB images and the sparse depth images. Finally, we compare against [96] (bottom row). Our proposed method, even using only 100 samples, outperforms [96] with 225 laser measurements. This is because our samples are spatially uniform, and thus provides more information than a line measurement. A few examples of our predictions with different inputs are displayed in Figure 3-2.
### Table 3.2: Comparison with state-of-the-art on the NYU-Depth-v2 dataset. The values are those originally reported by the authors in their respective paper

<table>
<thead>
<tr>
<th>Problem</th>
<th>#Samples</th>
<th>Method</th>
<th>rmse</th>
<th>REL</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB</td>
<td>0</td>
<td>Roy et al. [172]</td>
<td>0.744</td>
<td>0.187</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Eigen et al. [110]</td>
<td>0.641</td>
<td>0.158</td>
<td>76.9</td>
<td>95.0</td>
<td>98.8</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Laina et al. [111]</td>
<td>0.573</td>
<td><strong>0.127</strong></td>
<td><strong>81.1</strong></td>
<td>95.3</td>
<td>98.8</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Ours-RGB</td>
<td><strong>0.514</strong></td>
<td>0.143</td>
<td>81.0</td>
<td><strong>95.9</strong></td>
<td><strong>98.9</strong></td>
</tr>
<tr>
<td>d</td>
<td>20</td>
<td>Ours-d</td>
<td>0.461</td>
<td>0.110</td>
<td>87.2</td>
<td>96.1</td>
<td>98.8</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>Ours-d</td>
<td>0.347</td>
<td>0.076</td>
<td>92.8</td>
<td>98.2</td>
<td>99.5</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>Ours-d</td>
<td><strong>0.259</strong></td>
<td><strong>0.054</strong></td>
<td><strong>96.3</strong></td>
<td><strong>99.2</strong></td>
<td><strong>99.8</strong></td>
</tr>
<tr>
<td>RGBd</td>
<td>225</td>
<td>Liao et al. [96]</td>
<td>0.442</td>
<td>0.104</td>
<td>87.8</td>
<td>96.4</td>
<td>98.9</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>Ours-RGBd</td>
<td>0.351</td>
<td>0.078</td>
<td>92.8</td>
<td>98.4</td>
<td>99.6</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>Ours-RGBd</td>
<td>0.281</td>
<td>0.059</td>
<td>95.5</td>
<td>99.0</td>
<td>99.7</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>Ours-RGBd</td>
<td><strong>0.230</strong></td>
<td><strong>0.044</strong></td>
<td><strong>97.1</strong></td>
<td><strong>99.4</strong></td>
<td><strong>99.8</strong></td>
</tr>
</tbody>
</table>

### Table 3.3: Comparison with state-of-the-art on the KITTI dataset. The Make3D values are those reported in [8]

<table>
<thead>
<tr>
<th>Problem</th>
<th>#Samples</th>
<th>Method</th>
<th>rmse</th>
<th>REL</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB</td>
<td>0</td>
<td>Make3D [111]</td>
<td>8.734</td>
<td>0.280</td>
<td>60.1</td>
<td>82.0</td>
<td>92.6</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Mancini [116]</td>
<td>7.508</td>
<td>-</td>
<td>31.8</td>
<td>61.7</td>
<td>81.3</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Eigen et al. [8]</td>
<td>7.156</td>
<td><strong>0.190</strong></td>
<td><strong>69.2</strong></td>
<td>89.9</td>
<td><strong>96.7</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Ours-RGB</td>
<td><strong>6.266</strong></td>
<td>0.208</td>
<td>59.1</td>
<td><strong>90.0</strong></td>
<td>96.2</td>
</tr>
<tr>
<td>RGBd</td>
<td>~650</td>
<td>full-MAE [95]</td>
<td>7.14</td>
<td>0.179</td>
<td>70.9</td>
<td>88.8</td>
<td>95.6</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>Ours-RGBd</td>
<td>4.884</td>
<td>0.109</td>
<td>87.1</td>
<td>95.2</td>
<td>97.9</td>
</tr>
<tr>
<td></td>
<td>225</td>
<td>Liao et al. [96]</td>
<td>4.50</td>
<td>0.113</td>
<td>87.4</td>
<td>96.0</td>
<td>98.4</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>Ours-RGBd</td>
<td>4.303</td>
<td>0.095</td>
<td>90.0</td>
<td>96.3</td>
<td>98.3</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>Ours-RGBd</td>
<td>3.851</td>
<td>0.083</td>
<td>91.9</td>
<td>97.0</td>
<td>98.6</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>Ours-RGBd</td>
<td><strong>3.378</strong></td>
<td><strong>0.073</strong></td>
<td><strong>93.5</strong></td>
<td><strong>97.6</strong></td>
<td><strong>98.9</strong></td>
</tr>
</tbody>
</table>
Figure 3-2: Predictions on NYU-Depth-v2. From top to bottom: (a) RGB images; (b) RGB-based prediction; (c) \(d\) prediction with 200 and no RGB; (d) \(RGBd\) prediction with 200 sparse depth and RGB; (e) ground truth depth.
The KITTI dataset is more challenging for depth prediction, since the maximum distance is 100 meters as opposed to only 10 meters in the NYU-Depth-v2 dataset. A greater performance boost can be obtained from using our approach. Although the training and test data are not the same across different methods, the scenes are similar in the sense that they all come from the same sensor setup on a car and the data were collected during driving. We report the values from each work in Table 3.3.

The results in the first RGB group demonstrate that RGB-based depth prediction methods fail in outdoor scenarios, with a pixel-wise rmse of close to 7 meters. Note that we use sparsely labeled depth image projected from LiDAR, instead of dense disparity maps computed from stereo cameras as in [8]. In other words, we have a much smaller training dataset compared with [8, 116].

An additional 500 depth samples bring the rmse to 3.3 meters, a half of the RGB approach, and boosts δ₁ from only 59.1% to 93.5%. Our performance also compares favorably to other fusion techniques including [95, 96], and at the same time demands fewer samples.

3.2.3 On Number of Depth Samples

In this section, we explore the relation between the prediction accuracy and the number of available depth samples. We train a network for each different input size for optimal performance. We compare the performance for all three kinds of input data, including RGB, d, and RGBd. The performance of RGB-based depth prediction is independent of input sample size and is thus plotted as a horizontal line for benchmarking.

On the NYU-Depth-v2 dataset in Figure 3-3, the RGBd outperforms RGB with over 10 depth samples and the performance gap quickly increases with the number of samples. With a set of 100 samples, the rmse of RGBd decreases to around 25cm, half of RGB (51cm). The REL sees a larger improvement (from 0.15 to 0.05, reduced by two thirds). On one hand, the RGBd approach consistently outperforms d, which indicates that the learned model is indeed able to extract information not only from the sparse samples alone, but also from the colors. On the other hand, the performance gap between RGBd and d shrinks as the sample size increases. Both approaches perform equally well when sample size goes up to 1000, which accounts for less than 1.5% of the image pixels and is still a small number compared with the image size. This observation indicates that the information extracted from the sparse sample set dominates the prediction when the sample size is sufficiently large, and in this case the color cue becomes almost irrelevant.

The performance gain on the KITTI dataset is almost identical to NYU-Depth-v2, as shown in Figure 3-4. With 100 samples the rmse of RGBd decreases from 7 meters to a half, 3.5 meters. This is the same percentage of improvement as on the NYU-Depth-v2 dataset. Similarly, the REL is reduced from 0.21 to 0.07, again the same percentage of improvement as the NYU-Depth-v2.

On both datasets, the accuracy saturates as the number of depth samples increases. Additionally, the prediction has blurry boundaries even with many depth samples. We believe both phenomena can be attributed to the fact that fine details are lost in bottleneck network architectures. It remains further study if additional skip connections from encoders to decoders help improve performance.
Figure 3-3: Impact of number of depth sample on the prediction accuracy on the NYU-Depth-v2 dataset. Left column: lower is better; right column: higher is better.
Figure 3-4: Impact of number of depth sample on the prediction accuracy on the KITTI dataset. Left column: lower is better; right column: higher is better.
3.3 Application: Dense Map from Visual Odometry Features

In this section, we demonstrate a use case of our proposed method in sparse visual SLAM and visual inertial odometry (VIO). The best-performing algorithms for SLAM and VIO are usually sparse methods, which represent the environment with sparse 3D landmarks. Although sparse SLAM/VIO algorithms are robust and efficient, the output map is in the form of sparse point clouds and is not useful for other applications (e.g. motion planning).

![Figure 3-5: Application in sparse SLAM and visual inertial odometry (VIO) to create dense point clouds from sparse landmarks. (a) RGB (b) sparse landmarks (c) ground truth point cloud (d) prediction point cloud, created by stitching RGBd predictions from each frame.](image)

To demonstrate the effectiveness of our proposed methods, we implement a simple visual odometry (VO) algorithm with data from one of the test scenes in the NYU-Depth-v2 dataset. For simplicity, the absolute scale is derived from ground truth depth image of
the first frame. The 3D landmarks produced by VO are back-projected onto the RGB image space to create a sparse depth image. We use both RGB and sparse depth images as input for prediction. Only pixels within a trusted region, which we define as the convex hull on the pixel space formed by the input sparse depth samples, are preserved since they are well constrained and thus more reliable. Dense point clouds are then reconstructed from these reliable predictions, and are stitched together using the trajectory estimation from VIO.

The results are displayed in Figure 3-5. The prediction map resembles closely to the ground truth map, and is much denser than the sparse point cloud from VO. The major difference between our prediction and the ground truth is that the prediction map has few points on the white wall, where no feature is extracted or tracked by the VO. As a result, pixels corresponding to the white walls fall outside the trusted region and are thus removed.

3.4 Self-supervised Training Framework

In this section, we focus on training a depth-completion network without using dense ground truth depth annotations. Instead, we only require sequences of RGB and sparse depth data for training.

Most of existing work on depth completion relies on densely annotated ground truth for training. However, dense ground truth generally can not be captured directly from depth sensors due to their limitations, as is discussed in Section 1.1.1. Even the acquisition of semi-dense labels can be technically challenging. For instance, Uhrig et al. [89] created an annotated depth dataset by aggregating consecutive data frames using GPS, stereo vision, and additional manual inspection. However, this method is not easily scalable. Furthermore, it produces only semi-dense annotations (∼30% pixels) within the bottom half of the image.

Figure 3-6: An illustration of the self-supervised training framework, which requires only a sequence of color images and sparse depth images. White rectangles are variables, red is the depth network to be trained, blue are deterministic computational blocks (without learnable parameters), and green are loss functions.

In this section, we propose a model-based self-supervised training framework for depth completion. This framework requires only a synchronized sequence of color/intensity images from a monocular camera and sparse depth images from lidar. Consequently, the self-supervised framework does not rely on any additional sensors, manual labeling work,
or other learning-based algorithms as building blocks. Furthermore, this framework does not depend on any particular choice of neural network architectures. The self-supervised framework is illustrated in Figure 3-6. During training, the current data frame $\text{RGBd}_1$ and a nearby data frame $\text{RGB}_2$ are both used to provide supervision signals. However, at inference time, only the current frame $\text{RGBd}_1$ is needed as input to produce a depth prediction $\text{pred}_1$.

### 3.4.1 Sparse Depth Supervision

The sparse depth input $d_1$ itself can be used as a supervision signal. Specifically, we penalize the differences between network input and output on the set of pixels with known sparse depth, and thus encouraging an identity mapping on this set. This loss leads to higher accuracy, improved stability and faster convergence for training. The depth loss is defined as

$$
\mathcal{L}_{\text{depth}}(\text{pred}, d) = \left\| \mathbb{1}_{\{d > 0\}} \cdot (\text{pred} - d) \right\|_2^2. 
$$

(3.3)

Note that a denser ground truth (e.g., the 30% dense annotation from the KITTI depth completion benchmark [89]), if available, can also be used in place of the sparse input $d_1$.

### 3.4.2 Frame-to-Frame Pose Estimation

As an intermediate step towards the photometric loss, the relative pose between the current frame and the nearby frame needs to be computed. Prior work assumes either known transformations (e.g., stereo [119]) or the use of another learned neural network for pose estimation (e.g., [113]). In contrast, in this framework, we adopt a model-based approach for pose estimation, utilizing both $\text{RGB}$ and $d$.

Specifically, we solve the Perspective-n-Point (PnP) problem [173] to estimate the relative transformation $T_{1 \rightarrow 2}$ between the current frame 1 and the nearby frame 2, using matched feature correspondences extracted from $\text{RGBd}_1$ and $\text{RGB}_2$ respectively. Random sample consensus (RANSAC) [174] is also adopted in conjunction with PnP to improve robustness to outliers in feature matching. Compared to $\text{RGB}$-based estimation [113] which is up-to-scale, our estimation is scale-accurate and failure-aware (flag returned if no estimation is found).

### 3.4.3 Temporal-Consistency Photometric Loss

Given the relative transformation $T_{1 \rightarrow 2}$ and the current depth prediction $\text{pred}_1$, the nearby color image $\text{RGB}_2$ can be inversely warped to the current frame. Specifically, given the camera intrinsic matrix $K$, any pixel $p_1$ in the current frame 1 has the corresponding projection in frame 2 as $p_2 = KT_{1 \rightarrow 2}\text{pred}_1(p_1)K^{-1}p_1$. Consequently, we can create a synthetic color image using bilinear interpolation around the 4 immediate neighbors of $p_2$. In other words, for all pixels $p_1$:

$$
\text{warped}_1(p_1) = \text{bilinear}(\text{RGB}_2(KT_{1 \rightarrow 2}\text{pred}_1(p_1)K^{-1}p_1)).
$$

(3.4)
warped is similar to the current RGB₁ when the environment is static and there’s limited occlusion due to change of view point. Note that this photometric loss is made differentiable by the bilinear interpolation. Minimizing the photometric error reduces the depth prediction error, only when the depth prediction is close enough to the ground truth (i.e., when the projected point \( p₂ \) differs from the true correspondence by no more than 1 pixel). Therefore, a multi-scale strategy is applied to ensure \( \left\| p₂^{(s)} - p₁^{(s)} \right\|₁ < 1 \) on at least one scale \( s \). In addition, to avoid conflicts with the depth loss, the photometric loss is evaluated only on pixels without direct depth supervision. The final photometric loss is

\[
\mathcal{L}_{\text{photometric}}(\text{warped}₁, \text{RGB}₂) = \sum_{s \in S} \frac{1}{s} \left\| \left( p₂^{(s)} \right)_{d=0} - (\text{warped}₁^{(s)} - \text{RGB}₂^{(s)}) \right\|₁,
\]

where \( S \) is the set of all scaling factors, and \( (\cdot)^{(s)} \) represents image resizing (with average pooling) by a factor of \( s \). Losses at lower resolutions are weighted down by \( s \).

### 3.4.4 Depth Smoothness Loss

The photometric loss only measures the sum of all individual errors (i.e., color differences computed on each pixel independently) without any neighboring constraints. Consequently, minimizing the photometric loss alone usually results in an undesirable local optimum, where the depth pixels have incorrect values (despite having a low photometric error) and high discontinuity. To alleviate this issue, we add a third term to the loss functions in order to encourage smoothness of the depth predictions. Inspired by [7, 91, 113], we penalize \( \| \nabla² \text{pred}₁ \|₁ \), the \( \mathcal{L}_1 \) loss of the second-order derivatives of the depth predictions, to encourage piecewise-linear depth signal.

In summary, the final loss function for the entire self-supervised framework consists of 3 terms:

\[
\mathcal{L}_{\text{self}} = \mathcal{L}_{\text{depth}}(\text{pred}₁, d₁) + \beta₁ \mathcal{L}_{\text{photometric}}(\text{warped}₁, \text{RGB}₁) + \beta₂ \| \nabla² \text{pred}₁ \|₁ \quad (3.6)
\]

where \( \beta₁, \beta₂ \) are relative weightings. Empirically we set \( \beta₁ = 0.1 \) and \( \beta₂ = 0.1 \).

### 3.5 Self-supervised Results

In this section, we evaluate the self-supervised training framework described in Section 3.4 on the KITTI validation dataset. We compare 3 different training methods: using only photometric loss without sparse depth supervision, the complete self-supervised framework (i.e., photometric loss with sparse depth supervision), and the pure supervised method using the semi-dense annotations. The quantitative results are listed in Table 3.4. The self-supervised result produces \( \text{rmse} = 1384 \), which already outperforms some of the prior methods that were trained with semi-dense annotations, such as SparseConvs [89].

However, note that the true quality of depth predictions trained in a self-supervised fashion is probably underestimated by such evaluation metrics, since the “ground truth” itself is biased. Specifically, the evaluation ground truth is characterized by the same lim-
Table 3.4: Evaluation of the self-supervised framework on the validation set

<table>
<thead>
<tr>
<th>Training Method</th>
<th>rmse [mm]</th>
<th>mae [mm]</th>
<th>irmse [1/km]</th>
<th>imae [1/km]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photometric Loss Only</td>
<td>1901.16</td>
<td>658.13</td>
<td>5.85</td>
<td>2.62</td>
</tr>
<tr>
<td>Self-Supervised</td>
<td>1384.85</td>
<td>358.92</td>
<td>4.32</td>
<td>1.60</td>
</tr>
<tr>
<td>Supervised Learning</td>
<td>878.56</td>
<td>260.90</td>
<td>3.25</td>
<td>1.34</td>
</tr>
</tbody>
</table>

iterations as the training annotations: low-density, as well as absence at the top region. As a result, predictions at the top, where the self-supervised framework provides supervision but semi-dense annotations do not, are not reflected in the error metrics, as illustrated in Figure 3-7.

The self-supervised framework is effective for not only 64-line lidar measurements, but also lower-resolution lidars and more sparse depth input. In Figure 3-8(b), we show the validation errors of the networks trained with the self-supervised framework with different levels of sparsity in the depth. When the number of input measurements is too small, the validation error is high. This is expected due to failure in PnP pose estimation. However, with sufficiently many measurements (e.g., at least 4 scanlines, or the equivalent number of samples to at least 2 scanlines when input is uniformaly sampled), the validation error starts to decrease as a power function of the input, similar to training with semi-dense annotations.

![Figure 3-7: Comparison between different training methods (best viewed in color). The photometric loss provides supervision at the top, where the semi-dense annotation does not contain labels.](image)

3.5.1 On Input Sparsity

In many robotic applications, engineers need to address the following question: *what’s the lidar resolution (which translates to financial cost) required to achieve certain performance?* In this section, we try to answer this question by evaluating the accuracy of our
lidar depth completion technique under different input sparsity and spatial patterns. To this end, we provide an empirical analysis on the depth completion accuracy for different depth input with varying levels of sparsity and spatial patterns. In particular, we downsample the raw lidar input in two different manners: reducing the number of laser scans (to simulate a lidar with fewer scan lines), and uniformly sub-sampling from all lidar measurements available. The results are illustrated in Figure 3-8, for both of these spatial patterns and both input modalities of d and RGBd.

In Figure 3-8(a) we show the validation errors when trained with semi-dense annotations. The \( \text{rmse} \) errors form a straight line in the log-log plot, implying that the depth completion error decreases as a power function \( c x^p \) of the number of input depth measurements, for some positive \( c \) and negative \( p \). This also implies diminishing returns on increasing lidar resolutions. Comparing the two spatial patterns, uniform random sub-sampling produces significantly higher accuracy than having a reduced number of scan lines, since the input depth samples are more disperse in the pixel space with uniform random sampling. Furthermore, using RGBd substantially reduces prediction error, compared to using only d, when trained with semi-dense annotations. The performance gap is especially significant when the number of depth measurements is low. Note that there is a significant drop of RMSE from 32-line to 64-line lidar. This accuracy gain may be attributed to the fact that our network architecture is optimized for 64-line lidar.

In Figure 3-8(b), we show results when trained with our self-supervised framework. The validation error starts to decrease steadily as a power function, similar to training with semi-dense annotations, when there are sufficiently many input measurements. However, with the self-supervised framework, using both RGB and sparse depth yields the same level of accuracy as using sparse depth only, which is different from training with semi-dense annotations. The underlying cause of this difference remains to be further investigated\(^2\).

\(^2\)In the self-supervised framework, the training process is more iterative than training with semi-dense annotations. In particular, it takes many more iterations for the predictions to converge to the correct value. Consequently, the network weights for the RGB input, which has substantially lower correlation with the depth prediction than the sparse depth input, might have dropped to negligible levels during early iterations, resulting in similar performance for using d and RGBd as input. However, this conjecture remains to be verified.
Figure 3-8: Prediction error against number of input depth samples, for both spatial patterns (uniform random sub-sampling and lidar scan lines). (a) When trained with semi-dense ground truth, the depth completion error decreases as a power function $c x^p$ of the number of input depth measurements, for some $c > 0, p < 0$. (b) The self-supervised framework is effective with sufficiently many measurements (at least 4 scanlines, or the equivalent number of samples to 2 scanlines when input is uniformly sampled).
3.6 Summary

In this chapter, we have developed a deep regression model for depth completion of sparse lidar measurements. We demonstrated that this method of using both RGB and sparse depth for depth estimation significantly outperforms using only RGB images, as well as other existing RGB-D fusion techniques. The empirical result implies that depth completion is a relatively simple problem, since a handful of sparse depth samples as input leads to fairly accurate depth estimation results. In contrast, monocular depth prediction based solely on RGB images produces low accuracy, indicating that monocular depth prediction is indeed highly ill-posed. Our model achieves state-of-the-art performance on the KITTI depth completion benchmark, and outperforms existing published work by a significant margin at the time of submission.

This regression method for depth completion can be used as a plug-in module in sparse SLAM and visual inertial odometry algorithms, as well as in super-resolution of LiDAR measurements. We believe that this new method opens up an important avenue for research into RGB-D learning and the more general 3D perception problems, which might benefit substantially from sparse depth samples.

We also propose a highly scalable, model-based self-supervised training framework for depth completion networks. This framework requires only sequences of RGB and sparse depth images, and outperforms a number of existing solutions trained with semi-dense annotations. Additionally, we present empirical results demonstrating that depth completion errors decrease as a power function with the number of input depth measurements.

For future work, we are interested in developing techniques for improving the self-supervised framework, including better loss functions and taking dynamic objects into account.
Chapter 4

Exact Reconstruction from Network Inversion

In this chapter, we present our third contribution\(^1\): a novel approach that utilizes the representation power of deep neural networks (as in Chapter 3) for image modeling, but also comes with performance guarantees of optimization theories (as in Chapter 2). This third contribution lies in between Chapter 2 and Chapter 3 on the model-based vs. data-driven spectrum.

In particular, we assume that the image formation process (of both RGB and depth) can be well approximated by a generative neural network\(^2\). In other words, we assume that there exists a generative neural network \(G\) and some low-dimensional latent representation \(z\), such that a pair of RGB and depth images of the same scene (denoted as \(x = [\text{RGB}, \text{depth}]\)) can be represented as \(x = G(z)\). Under such an assumption, the depth estimation problem can be solved by addressing a more generic non-convex optimization problem: given partial measurements \(y\) of the network output \(x\), find the underlying input latent code \(z\). Once the latent code \(z\) is found, the complete network output can be recovered as \(G(z)\). This problem can be understood more intuitively as “compressive sensing on deep neural networks”.

4.1 Network Inversion and Related Work

In recent years, generative models have made significant progress in learning representations for complex and multi-modal data distributions, such as those of natural images [176, 177]. However, despite the empirical success, there has been very little theoretical understanding into the mapping itself from the input latent space to the high-dimensional space.

For instance, *generative adversarial networks* (GAN) [177, 178, 179] and *variational auto-encoders* (VAE) [176] use deep neural networks to learn the mapping from a low-dimensional representation space to a high dimensional sample space. It is worth noting that the state-of-the-art generative models [180] use deep convolutional networks rather

---

\(^1\)The work presented in this chapter has been published [175].

\(^2\)A generative neural network takes a low-dimensional vector as input, and generates a high-dimensional, photo-realistic image as output. Many solutions (e.g., GAN) have been proposed to train such networks and have been demonstrated to perform well.
than fully-connected networks. These methods found success in producing realistic looking images. In this chapter, we address the following question: given a convolutional generative network\(^3\), is it possible to “decode” an output image and recover the corresponding input latent code? In other words, we are interested in the \textit{invertibility of convolutional generative models}.

The answers to such questions not only provide theoretical insights into convolutional generative networks, but also create a new paradigm for many challenging inference problems such as image classification. For instance, one can label an image simply by inverting a conditional generative model whose latent space has clear physical interpretations that relate to the labels. Another example is the case where only a small portion of the generative model’s output is observed, as illustrated in Figure 4-1, which found applications in image reconstruction and image inpainting [137, 181].

\[
\text{Figure 4-1: Recovery of the input latent code } z \text{ from under-sampled measurements } y = AG(z) \text{ where } A \text{ is a sub-sampling matrix and } G \text{ is an expanding generative neural network. We prove that } z \text{ can be recovered with guarantees using simple gradient-descent methods under mild technical assumptions. Best viewed in color.}
\]

This inversion problem is interesting for many reasons. Firstly, it provides insight into the mapping from the low-dimensional latent space to the high-dimensional image space (\textit{e.g.}, is the mapping one-to-one or many-to-one?). A deeper understanding of the mapping can potentially help solve the well known mode collapse\(^4\) problem [182] during the training in the generative adversarial network (GAN) [178, 179]. Secondly, the inversion itself can be applied in image manipulation [183] (\textit{e.g.}, vector arithmetic of face images [184]), image in-painting [137, 181], and image reconstruction from sparse measurements [7, 88].

\(^3\)Deep generative models typically use transposed convolution (\textit{a.k.a.} “deconvolution”). With a slight abuse of notation we refer to transposed convolutional generative models as convolutional models.

\(^4\)Mode collapse refers to the problem that the Generator characterizes only a few images to fool the discriminator in GAN. In other words, the input latent space is mapped to only a few points in the image space.
The challenge of the inversion of a deep neural network lies in the fact that the inversion problem is highly non-convex, which implies that gradient-based methods do not have performance guarantees. However, in this work, we show that the landscape of empirical risk has a favorable geometry, despite being non-convex. Specifically, we prove that with simple first-order algorithms like stochastic gradient descent and its variants, we can recover the latent code with guarantees.

The inversion of a deep neural network, as well as the optimization of its weights, is a highly non-convex problem. Despite the non-convex nature of the objective functions, simple first-order algorithms like stochastic gradient descent and its variants often produce desirable results. This is because usually the landscape of objective functions usually have a favorable geometry despite being non-convex, such as having no spurious local minima. Also there are recent works in solving non-convex inverse problems [185, 186].

**Related Work**

Bora et al. [187] empirically find that minimizing the non-convex Problem 4.3, which is defined formally in Section 4.2, using standard gradient-based optimizer yields good reconstruction result from a small number of Gaussian random measurements. They also provide guarantees on the global minimum of a generative network with certain structure. However, their work does not analyze how to find the global minimum.

Hand and Voroninski [188] further establish that a fully connected generative network with weights following Gaussian distribution can be inverted given only compressive linear observations of its last layer. In particular, they show that under mild technical conditions Problem 4.3 has a favorable global geometry, in the sense that there are no stationary points outside of neighborhoods around the desired solution and its negative multiple with high probability. However in the generative models that are used in practice, mostly deconvolutional networks are mostly used rather fully connected ones. This motivates us to study the landscape of deconvolutional neural networks and show theoretical guarantees for gradient descent approach to find the global minimum.

We also note the work [189], which provides a theoretical connection between model-based compressed sensing and the invertibility of convolutional neural networks. However, their assumption on the input signal is different than ours, and their methods appear not able to handle nonlinearities such as ReLUs.

Another line of research, which focuses on gradient-based algorithms, analyzes the behavior of (stochastic) gradient descent for Gaussian-distributed input. Soltanolkotabi [190] showed that projected gradient descent is able to find the true weight vector for 1-layer, 1-neuron model. More recently, Du et al. [191] improved this result for a simple convolutional neural network with two unknown layers. Their assumptions on random input and their problem of weight learning are different than the problem we study in this chapter.

Our problem is also connected to compressive sensing [192, 193] which exploits the sparsity of natural signals to design acquisition schemes where the number of measurements scales linearly with the sparsity level. In this work, we go beyond this sparsity assumption. Specifically, we exploit the hierarchical nature of images and other natural signals by leveraging the powerful representation capability of deep learning.

**Contribution**  
In this chapter we make two main contributions:
• We prove that a convolutional generative neural network is invertible, even when only partial output is observed, under the following assumptions: (1) the network consists of two layers of transposed convolutions followed by ReLU activation functions; (2) the network is (sufficiently) expansive; (3) the filter weights follow a Gaussian distribution. When these conditions are satisfied, the input latent code can be recovered from partial output of a generative neural network by minimizing a $\mathcal{L}_2$ empirical loss function using gradient descent. We discuss how we can extend this theorem to multiple networks and Leaky ReLU, but defer them to future work.

• We empirically validate our theoretical results by reconstructing fake and real images from partial observations. This is essentially inpainting problem and our experiments show that our theoretical results generalize to (1) multiple-layer networks; (2) networks with other nonlinear activation functions, including Leaky ReLU, Sigmoid and Tanh.

Two key contributions of our proof includes (a) the concentration bounds of weight matrices combined with ReLU operation, and (b) the angle distortion between two arbitrary input vectors under the transposed convolution and ReLU. In general, our proof follows a similar basic structure to [188], where they show the invertibility of fully connected networks with Gaussian weights. However, in fully connected networks, the weight matrix of each layer is a dense Gaussian matrix. In contrast, in convolutional networks the weight matrices are highly sparse with block structure due to striding filters, as in Figure 4-2(a). Therefore, [188]'s proof doesn't apply to convolutional networks, and the extension of concentration bounds for our case is not trivial.

To address such problem, we propose a new permutation technique which shuffles the rows and columns of weight matrices to obtain a block matrix, as in Figure 4-2(b). With permutation, each block is now a dense Gaussian matrix, where we can apply existing matrix concentration results. However, the permutation operation is quite arbitrary, depending on the structure of the convolutional network. This requires some careful handling, since the second step (b) requires the control of angles.

In addition, Hand and Voroninski [188] assume a Gaussian sub-sampling matrix at the output of the network, rather than partial sub-sampling (sub-matrix of identity matrix) that we study in this problem. We observe that the sub-sampling operation can be swapped with the last ReLU in the network, since both are pointwise operations. We handle the sub-sampling by making the last layer a bit more expansive, and prove that it is the same with no downsampling from a theoretical standpoint.

### 4.2 Assumptions and Notation

In this section, we introduce the notation and define the network inversion problem. Let $z^0 \in \mathbb{R}^{n_0}$ denote the latent code of interest, $G(\cdot) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_d}$ ($n_0 \ll n_d$) be a $d$-layer generative network that maps from the latent space to the image space. Then the ground truth output image $x^0 \in \mathbb{R}^{n_d}$ is produced by

$$x^0 = G(z^0),$$

(4.1)
In this chapter we consider \( G(\cdot) \) to be a deep neural network\(^5\). In particular we assume \( G(\cdot) \) to be a two-layer transposed convolutional network, modeled by

\[
G(z) = \sigma(W_2 \sigma(W_1 z)) \tag{4.2}
\]

where \( \sigma(z) = \max(z, 0) \) denotes the rectified linear unit (ReLU) that applies entrywise. \( W_1 \in \mathbb{R}^{n_1 \times n_0} \) and \( W_2 \in \mathbb{R}^{n_2 \times n_1} \), are the weight matrices of the convolutional neural network in the first and second layers, respectively. Note that since \( G(\cdot) \) is a convolutional network, \( W_1 \) and \( W_2 \) are highly sparse with a particular block structure, as illustrated in Figure 4-2(a).

Let us make the inversion problem a bit more general by assuming that we only have partial observations of the output image pixels. Specifically, let \( A \in \mathbb{R}^{m \times n_2} \) be a sub-sampling matrix (a subset of the rows of an identity matrix), and then the observed pixels are \( y^o = Ax^o \in \mathbb{R}^m \). Consequently, the inversion problem given partial measurements can be described as follows:

Let: \( z^o \in \mathbb{R}^{n_0}, W_1 \in \mathbb{R}^{n_1 \times n_0}, W_2 \in \mathbb{R}^{n_2 \times n_1}, A \in \mathbb{R}^{m \times n_2} \)

Given: \( A, W_1, W_2 \) and observations \( y^o = AG(z^o) \)

Find: \( z^o \) and \( x^o = G(z^o) \)

Since \( x^o \) is determined completely by the latent representation \( z^o \), we only need to find \( z^o \). We propose to solve the following optimization problem for an estimate \( \hat{z} \):

\[
\hat{z} = \arg \min_z J(z), \text{ where } J(z) = \frac{1}{2} \| y^o - AG(z) \|^2 \tag{4.3}
\]

This minimization problem is highly non-convex because of \( G \). Therefore, in general a gradient descent approach is not guaranteed to find the global minimum \( z^o \), where \( J(z^o) = 0 \).

**Network Assumptions**

We vectorize the input signal to 1D signal. The feature at the \( i \)th layer consists of \( C_i \) channels, each of size \( D_i \). Therefore, \( n_i = C_i \cdot D_i \). At any convolutional layer, let \( f_{i,j} \) denote the kernel filter (each of size \( \ell \)) for the \( i \)th input channel and the \( j \)th output channel. For simplicity, we assume the stride to be equal to the kernel size \( \ell \). All filters can be concatenated to form a large block matrix \( W_i \). For instance, an example of such block matrix \( W_1 \) for the first layer is shown in Figure 4-2(a). Under our assumptions, the input and output sizes at each deconvolution operation can be associated as \( D_{i+1} = D_i \ell \).

Let \( D_v J(z) \) be one-sided directional derivative of the objective function \( J(\cdot) \) along the direction \( v \), *i.e.*, \( D_v J(z) = \lim_{t \to 0^+} \frac{J(z+tv) - J(z)}{t} \). Let \( B(z, r) \) be the Euclidean ball of radius \( r \) centered at \( z \). We omit some universal constants in the inequalities and use \( \gtrsim \) (if the constant depends on a variable \( \epsilon \)) instead.

\(^5\)Note that this network inversion problem happens at the inference stage, and thus is independent of the training process.
\[ f_{i,j} \text{ stands for the } i^{th} \text{ filter kernel for the } j^{th} \text{ input channel.} \]

\[ z \] and \[ x \] denote the input and output signals, respectively. (a) The standard transposed convolution represented as linear multiplication. (b) With proper row and column permutations, the permuted weight matrix has a repeating block structure.

### 4.3 Statement of Results

In this section, we present our main theoretical results regarding the invertibility of a 2-layer convolutional generative network with ReLUs. Our first main theoretical contribution is as follows: although the problem in (4.3) is non-convex, under appropriate conditions there is a strict descent direction everywhere, except in the neighborhood of \( z^\circ \) and that of a negative multiple of \( z^\circ \).

**Theorem 27 (Approximate invertibility of convolutional generative networks).** Fix \( \epsilon > 0 \). Let \( W_1 \in \mathbb{R}^{C_0D_0 \times C_1D_1} \) and \( W_2 \in \mathbb{R}^{C_1D_1 \times C_2D_2} \) be deconvolutional weight matrices with filters in \( \mathbb{R}^\ell \) with i.i.d. entries from \( \mathcal{N}(0, 1/C_i\ell) \) for layers \( i = 1, 2 \) respectively. Let the sampling matrix \( A = \text{Id} \) be an identity matrix (meaning there’s no sub-sampling). If \( C_1\ell \gtrsim \epsilon C_0 \log C_0 \) and \( C_2\ell \gtrsim \epsilon C_1 \log C_1 \) then with probability at least \( 1 - \kappa(D_1C_1e^{-\gamma C_0} + D_2C_2e^{-\gamma C_1}) \) we have the following. For all nonzero \( z \) and \( z^\circ \), there exists \( v_{z,z^\circ} \in \mathbb{R}^{n_0} \) such that

\[ D_{v_{z,z^\circ}} J(z) < 0, \quad \forall z \notin \mathcal{B}(z^\circ, \epsilon \|z^\circ\|_2) \cup \mathcal{B}(-\rho z^\circ, \epsilon \|z^\circ\|_2) \cup \{0\} \tag{4.4} \]

\[ D_z J(0) < 0, \quad \forall z \neq 0, \tag{4.5} \]

where \( \rho \) is a positive constant. Both \( \gamma > 0 \) and \( \kappa > 0 \) depend only on \( \epsilon \).

Theorem 27 establishes under some conditions that the landscape of the cost function is not adversarial. Despite the heavily loaded notation, Theorem 27 simply requires that
the weight matrices with Gaussian filters should be sufficiently expansive (i.e., output dimension of each layer should increase by at least a logarithmic factor). Theorem 27 does not provide information regarding the neighborhood centered at $-\rho x^o$, which implies the possible existence of a local minimum or a saddle point. However, empirically we did not observe convergence to a point other than the ground truth. In other words, gradient descent seems to always find the global minimum, see Figure 4-4. An implication of Theorem 27 is that the mapping from the latent code space to the high-dimensional image space is approximately one-to-one, with high probability.

One assumption we make is the size of stride $s$ being same as the filter size $\ell$. Although theoretically convenient, this assumption is not common in the practical choices of transposed convolutional networks. We believe a further analysis can remove this assumption, which we also leave as a future work. In practice different activation functions other than ReLU can be used as well, such as sigmoid function, Tanh and Leaky ReLU. It is also an interesting venue of research to see whether a similar analysis can be done with those activations. In particular, for Leaky ReLU we briefly explain how the proof would divert from ours in Section B.2. We include landscapes of the cost function when different activations are used in Figure 4-4.

Gaussian weight assumption might seem unrealistic at first. However, there is some research [194] indicating that weights of some trained networks follow a normal distribution. We also make a similar observation on the networks we trained, see Section 4.4. We also note that Theorem 27 does not require independence of network weights across layers.

**Proof Outline:** Due to space limitations, the complete proof of Theorem 27 is given in Appendix B. Here we give a brief outline of the proof and highlight the main steps. The theorem is proven by showing two main conditions on the weight matrices.

The first condition is on the spatial arrangement of the network weights within each layer. Lemma 35 provides a concentration bound on the distribution of the effective weight matrices (after merging the ReLUs into the matrices). It shows that the set of neuron weights within each layer are distributed approximately like Gaussian. A key idea for the proving Lemma 35 is our new permutation technique. Specifically, we rearrange both rows and columns of the sparse weight matrices, as in Figure 4-2(a), into a block diagonal matrix, as in Figure 4-2(b). Each block in the permuted matrix is the same Gaussian matrix with independent entries. The permutation into block matrices helps turns each block in Figure 4-2(b) into a dense Gaussian matrix, and therefore makes it possible to utilize existing concentration bounds on Gaussian matrices.

The second condition is on the approximate angle contraction property of an effective weight matrix $W_i$ (after merging the ReLUs into the matrices). Lemma 37 shows that the angle between two arbitrary input vectors $x$ and $y$ does not vanish under a transposed convolution layer and the ReLU. The permutation poses a significant challenge on the proof of Lemma 37, since permutation of the input vectors distorts the angles. The difficulty is handled carefully in the proof of Lemma 37, which deviates from the proof machinery in [188] and hence is a major technical contribution. □

**Conjecture 28.** Under the assumptions of Theorem 27, let the network weights follow any zero-mean subgaussian distribution $\mathbb{P}(|x| > t) \leq ce^{-\gamma t^2}$, $\forall t > 0$ instead of Gaussian.
Then with high probability the same conclusion holds.

A subgaussian distribution (a.k.a. light-tailed distribution) is one whose tail decays at least as fast as a Gaussian distribution (i.e., exponential decay). This includes, for example, any bounded distribution and the exponential distribution. Empirically, we observe that Theorem 27 holds for a number of zero-mean subgaussian distributions, including uniform random weights and \{+1, −1\} binary random weights.

Now let us move on to the case where the subsampling matrix \(A\) is not an identity matrix. Instead, consider a fixed sampling rate \(r \in (0, 1]\).

**Theorem 29** (Approximate invertibility under partial measurements). Under the assumptions of Theorem 27, let \(r \in (0, 1]\) be a constant sampling ratio and \(A \in \mathbb{R}^{m \times C_2 D_2}\) be an arbitrary subsampling matrix with \(m/(C_2 D_2) \geq r\). Then with high probability the same result as Theorem 27 holds.

Note that the subsampling rate \(r\) appears in the dimension of the weight matrix of the second layer.

**Proof.** Since ReLU operation is pointwise, we have the identity

\[
y = AG(z) = A\sigma(W_2\sigma(W_1z)) = \sigma(AW_2\sigma(W_1z)).
\]

It suffices to show that Theorem 27 still holds with \(AW_2\) as the last weight matrix. Note that \(AW_2\) selects a row subset of the matrix \(W_2\) Figure 4-2(a). Consequently, after proper permutation, \(AW_2\) is again a block diagonal matrix with each block being a Gaussian matrix with independent entries. Only this time the blocks are not identical, but instead have different sizes. As a result, Theorem 27 still holds for \(AW_2\), since the proof of Theorem 27 does not require the identical blocks. However, there are certain dimension constraints, which can be met by expanding the last layer with a factor of \(r\), the sampling rate. This modification is reflected in the additional dimension assumption on the weight matrix \(W_2\). \(\square\)

The minimal sampling rate \(r\) is a constant that depends on both the network architecture (e.g., how expansive the networks are) and the sampling matrix \(A\). We made 2 empirical observations. Firstly, spatially disperse sampling patterns (e.g., uniform random samples) require a lower \(r\), whilst more aggressive sampling patterns (e.g., top half, left half, sampling around image boundaries) demand more measurements for exact recovery. Secondly, regardless of the sampling patterns \(A\), the probability of exact recovery exhibits a phase transition phenomenon w.r.t. the sampling rate \(r\). This observation supports Theorem 4 (i.e., network is invertible given sufficient measurements). A more rigorous and mathematical characterization of \(r\) remains an open question.

### 4.4 Experimental Validation

In this section, we verify the gaussian weight assumption of trained generative networks, our main result Theorem 29 on simulated 2-layer networks, as well as the generalization of Theorem 29 to more complex multi-layer networks trained on real datasets.
4.4.1 Gaussian Weight in Trained Networks

Figure 4-3: Distribution of the kernel weights from every layer in a trained convolutional generative network. The trained weights roughly follow a zero-mean gaussian distribution.

We extract the convolutional filter weights, trained on real data to generate images in Figure 4-5, from a 4-layer convolutional generative models. The histogram of the weights in each layer is depicted in Figure 4-3. It can be observed that the trained weights highly resembles a zero-mean gaussian distribution. We also discover similar distributions of weights in other trained convolutional networks, such as ResNet [6]. Arora et al. [194] also report similar results.

4.4.2 On 2-layer Networks with Random Weights

As a sanity check on Theorem 29, we construct a generative neural network with 2 transposed convolution layers, each followed by a ReLU. The first layer has 16 channels and the second layer has 1 single channel. Both layers have a kernel size of 5 and a stride of 3. In order to be able to visualize the cost function landscape, we set the input latent space to be 2-dimensional. The weights of the transposed convolution kernels are drawn i.i.d. from a Gaussian distribution with zero mean and unit standard deviation. Only 50% of the network output is observed. We compute the cost function $J(z)$ for every input latent code $z$ on a grid centered at the ground truth. The landscape of the cost function $J(z)$ is depicted in Figure 4-4(a). Although Theorem 29 implies a possibility of a stationary point at the negative multiple of the ground truth, experimentally we do not observe convergence to any point other than the global minimum.

Despite the fact that Theorem 27 and Theorem 29 are proved only for the case of 2-layer network with ReLU, the same conclusion empirically extends to networks with more layers and different kernel sizes and strides. In addition, the inversion of generative models generalizes to other standard activation functions including Sigmoid, and Tanh. Specifically, Sigmoid and Tanh have quasi-convex landscapes as shown in Figure 4-4(b) and (c),
which are even more favorable than that of ReLU. Leaky ReLU has the same landscape as a regular ReLU.

Figure 4-4: The landscape of the cost function $J(z)$ for deconvolutional networks with (a) ReLU, (b) Sigmoid, and (c) Tanh as activation functions, respectively. There exists a unique global minimum.

As a counter example, we draw kernel weights uniformly randomly from $[0, 1]$ (which violates the zero-mean Gaussian assumption). Consequently, there is a flat global minimum in the latent space, as shown in Figure 4-4(d). In this region, any two latent vectors are mapped to the exact same output, indicating that mode collapse indeed occurs.

### 4.4.3 On Multi-layer Networks Trained with Real Data

In this section, we demonstrate empirically that our finding holds for multi-layer networks trained on real data. The first network is trained with GAN to generate handwritten digits, and the second for celebrity faces. In both experiments, the correct latent codes can be recovered exactly from partial (but sufficiently many) observations.
**MNIST:** For the first network on handwritten digit, we rescale the raw grayscale images from the MNIST dataset [195] to size of $32 \times 32$. We used the conditional deep convolutional generative adversarial networks (DCGAN) framework [177, 179] to train both a generative model and a discriminator. Specifically, the generative network has 4 transposed convolutional layers. The first 3 transposed convolutional layers are followed by a batch normalization and a Leaky ReLU. The last layer is followed by a Tanh. The discriminator has 4 convolutional layers, with the first 3 followed by batch normalization and Leaky ReLU and the last one followed by a Sigmoid function. We use Adam with learn rate $0.1$ to optimize the latent code $z$. The optimization process usually converges within 500 iterations. The input noise to the generator is set to have a relatively small dimension 10 to ensure a sufficiently expanding network.

![Image](image.png)

Figure 4-5: We demonstrate recovery of latent codes on a generative network trained on the MNIST dataset. From top to bottom: ground truth output images; partial measurements with different sampling masks; reconstructed image using the recovered latent codes from partial measurements. The recovery of latent codes is exact using simple gradient descent.

5 different sampling matrices are showcased in Figure 4-5, including observing uniform random samples, as well as the top half, bottom half, left half, and right half of the image space. In all cases, the input latent codes are recovery exactly. We feed the recovered latent code as input to the network to obtain the completed image, shown in the $3^{rd}$ row.

**CelebFaces:** A similar study is conducted on a generative network trained on the CelebFaces [184] dataset. We rescale the raw grayscale images from the CelebFaces dataset to size of $64 \times 64$. A similar network architecture to previous MNIST experiment is adopted, but both the generative model and the discriminator have 4 layers rather than 3. The images are showcased in Figure 4-6.

Note that the probability of exact recovery increases with the number of measurements. The minimum number of measurements required for exact recovery, however, depends on the network architecture, the weights, and the sampling spatial patterns. The mathematical characterization for minimal number of measurements remains a challenging open question.
Figure 4-6: recovery of latent codes on a generative network trained on the CelebA dataset. From top to bottom: ground truth output images; partial measurements with different sampling masks; reconstructed image using the recovered latent codes from partial measurements. The recovery of latent codes is exact using simple gradient descent.

4.5 Summary

In this chapter we prove rigorously that a 2-layer ReLU convolutional generative neural network is invertible, even when only partial output is observed. This result provides a sufficient condition for the generator network to be approximately one-to-one, which avoids the mode collapse problem in training of GAN. We empirically demonstrate that the same conclusion holds even if the generative models have other nonlinear activation functions (LeakyReLU, Sigmoid and Tanh) and multiple layers. The same proof technique can be potentially generalized to multi-layer networks. Some interesting future research directions include rigorous proofs for leaky ReLUs and other activation functions, subgaussian network weights, as well as inversion under noisy measurements.

The two-stage algorithm developed assumed that the target image is the output of a given generative neural network. This assumption is currently not satisfied for practical purposes, since there is no guarantee that a real image lies in the range of a given neural network. Therefore, it is of both practical and theoretical interests to extend the results to noisy measurements, where the assumption is relaxed and the target image only needs to be close to the range of the network.
Chapter 5

Conclusions and Remarks

In this thesis, we addressed the problem of depth image reconstruction from sparse depth measurements from both the algorithmic and computational perspectives. On the model-based vs. data-driven methodological spectrum, we presented three different algorithms: a pure model-based approach (Chapter 2), a pure data-driven approach (Chapter 3), and an approach that marries the merits of both worlds (Chapter 4) when assumptions are satisfied.

In Chapter 2, we propose a new approach to recover dense 2D and 3D depth signals from sparse and incomplete depth measurements. As a first contribution, we formulate depth reconstruction as the problem of finding a signal that has the sparsest second-order derivative, i.e., the least amount of corners and edges, while matching the given measurements. The problem itself is NP-hard, hence we relax it to a convex $\ell_1$-minimization problem with $\ell_\infty$-norm constraints. Additionally, we developed a fast solver called NESTA that outperforms generic linear programming solvers in terms of speed. A set of technical conditions under which exact reconstruction of the planar structures have been studied, along with rigorous proofs and analysis. We demonstrated empirically that this algorithm outperforms naive interpolation-based approach, as well as deep-learning-based methods, especially if the sampling is uniform and the scene geometry is indeed planar. Violation of these assumptions, such as biased sampling and complex geometry in outdoor settings, leads to low accuracy.

In Chapter 3, we presented a deep regression network for depth completion, as well as a self-supervised learning framework that trains such network without requiring dense ground truth. We have demonstrated empirically that this supervised approach achieves state of the art accuracy on the KITTI dataset. In particular, the quality of the estimated depth improves significantly with a small number of sparse depth measurements, when compared against depth prediction based on color images alone. The depth estimation is generally accurate, with the exception of object boundaries which tend to be over-blurry. The self-supervised learning framework trains the same network without using ground truth dense depth annotations, but instead enforces temporal photometric consistency using only video streams and sparse lidar data. This deep-learning approach is less sensitive to biased sampling patterns (such as Lidar scanlines at the bottom of image space) and does not rely on specific assumptions of scene geometry. However, this data-driven approach implicitly requires a similar distributions between training and testing datasets, and might observe sub-optimal accuracy when deployed to a vastly different scene than the ones in the training
dataset.

We also proposed an algorithm for exact reconstruction of depth (as well as color) images in Chapter 4, under the assumption that the target image is the output of a given generative neural network. We proved that the reconstruction strategy is guaranteed to converge to the global optimum under a set of technical assumptions (2-layer transposed-convolutional networks with ReLU activations and Gaussian random weights). Empirically we also demonstrated that the proposed algorithm works with high success rate on deeper neural networks trained with real data. Unfortunately, the assumption that any image lies in the range of a given generative neural network is usually not satisfied in practice. Therefore, this approach yield lower accuracy than the first two methods for a real depth image, despite the fact that it achieves perfect reconstruction for a synthetic image that is generated from the network.

Our results demonstrated that dense depth reconstruction from sparse depth measurements is feasible. However, these algorithms yield different levels of accuracies, depending on the sampling patterns as well as whether the scene geometry aligns with the modeling assumptions. These assumptions need to be assessed prior to choosing and deploying the algorithms.

5.1 Future directions

There are several limitations with the approaches developed in this dissertation. These algorithmic limitations could be a potential barrier for practical applications, and thus require further research efforts.

The regression-based algorithms presented in Chapter 3 empirically produce over-smooth boundaries between objects, which can be attributed to the fact that the network output is generated from convolutional operations and that neighboring pixels have similar receptive fields. This is a fundamental problem in convolutional neural networks. In addition, there is no guarantees that the predicted depth align with the sparse depth input. Enforcing this input-output consistency could potentially lead to significant accuracy enhancement and algorithmic simplifications.

The two-stage algorithm developed in Chapter 4 assumed that the target image is the output of a given generative neural network. This assumption is currently not satisfied for practical purposes, since there is no guarantee that a real image lies in the range of a given neural network. Therefore, it is of both practical and theoretical interests to extend the results to noisy measurements, where the assumption is relaxed and the target image only needs to be close to the range of the network.

The algorithms we have presented in this thesis demonstrate that dense depth image reconstruction from sparse measurements is feasible and can be applied in different tasks such as dense mapping. However, all three proposed algorithms rely on different set of assumptions that might be hard to validate for practical purposes. A more generic algorithm that makes weaker assumption regarding the target image would lead to broader adoption of depth reconstruction techniques.

For robotic and other safety-critical applications, both runtime and accuracies are of utmost importance. A future direction is to improve quality of 3D reconstruction and to
reduce average runtime by fusing single-view depth estimation across several time frames. It is also possible to utilize other sources of information, such as object and semantic understanding, to help improve the depth quality.
Appendix A

Proofs and Analysis for Chapter 2

A.1 Some Useful Lemmas

We introduce some technical lemmas which simplify the derivations in the following sections.

**Lemma 30** (Null space of $A$). Consider the sparse sampling matrix $A \in \mathbb{R}^{m \times n}$. The null space of $A$ is spanned by the rows of the matrix $N = I_{\overline{\mathcal{M}}} \in \mathbb{R}^{\bar{m} \times n}$ (with $\bar{m} = n - m$). Moreover, the action of the matrix $N$ on a vector $v$ and on a matrix $V$ of suitable dimensions is such that $Nv = v_{\overline{\mathcal{M}}}$ and $NV = V_{\overline{\mathcal{M}}}$.

**Proof.** Denote the $i$-th standard basis vector as $e_i$. Each row of $A \in \mathbb{R}^{m \times n}$ is equal to $e_i^T$ for some $i \in \mathcal{M}$, hence $A$ has rank $m$. Since the sets $\mathcal{M}$ and $\overline{\mathcal{M}}$ are disjoint and are such that $\mathcal{M} \cup \overline{\mathcal{M}} = \{1, \ldots, n\}$, it follows that $AN^T = 0$ (entries of $AN^T$ have the form $e_i^T e_j$ which is zero for $\mathcal{M} \ni i \neq j \in \overline{\mathcal{M}}$) and $N^T$ has rank $\bar{m} = n - m$. This proves that the rows of $N$ span the null space of $A$. Since each row of $N$ is $e_j^T$ for some $j \in \overline{\mathcal{M}}$, the claims $Nv = v_{\overline{\mathcal{M}}}$ and $NV = V_{\overline{\mathcal{M}}}$ easily follow. \qed

**Lemma 31** (Symmetric Tridiagonal Toeplitz matrix). Let $T$ denote a symmetric tridiagonal Toeplitz matrix with diagonal entries equal to $-2$ and off-diagonal entries equal to 1:

$$ T = \begin{bmatrix} -2 & 1 & 0 & 0 & \ldots & 0 \\ 1 & -2 & 1 & 0 & \ldots & 0 \\ 0 & 1 & -2 & 1 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 1 \\ 0 & \ldots & 0 & 0 & 1 & -2 \end{bmatrix} \quad (A.1) $$

Then the following claims hold:

(i) $T$ is invertible;

(ii) all the entries in the first and in the last column of $T^{-1}$ are negative and have absolute value smaller than 1;

(iii) let $v \in \mathbb{R}^n$ be defined as $v = [1 \ 0 \ \ldots \ 0 \ 1]^T$, then $T^{-1}v = -1$. 

111
Proof. Invertibility follows from [196, Corollary 4.2], which also reports the explicit form of the inverse of a Toeplitz matrix. We report the inverse here, tailoring it to our matrix. For the \(n \times n\) Toeplitz matrix \(T\) in eq. (A.1), the entry in row \(i\) and column \(j\) of \(T^{-1}\) is:

\[
(T^{-1})_{ij} = \begin{cases} 
(-1)^{2i-1} \frac{i(n-j+1)}{n+1} & \text{if } i \leq j \\
(-1)^{2j-1} \frac{j(n-i+1)}{n+1} & \text{if } i > j 
\end{cases}
\]  

(A.2)

By inspection one can see that the first column \((j = 1)\) and the last column \(j = n\) are all negative and have absolute value smaller than 1. The last claim can be proven by observing that \(T1 = -v\) and the matrix is invertible.

\[\square\]

Lemma 32 (Null Space of D). Given a 2\textsuperscript{nd}-order difference operator \(D \in \mathbb{R}^{(n-2) \times n}\), defined as in (2.6), the null space of \(D\) is spanned by the following vectors:

\[v_1 = 1_n, \quad v_2 = \begin{bmatrix} 1 & 2 & \cdots & n \end{bmatrix}^T\]  

(A.3)

Proof. By inspection one can see that \(Dv_1 = Dv_2 = 0\). Moreover, the rank of \(D\) is \(n-2\) and \(v_1\) and \(v_2\) are two linearly independent vectors, which proves the claim.

\[\square\]

A.2 Proof of Proposition 7

In this appendix we prove that in 2D depth reconstruction problems, if we sample only the corners of the signal (and the first and the last entry), then \(C_{er} = 1\). Moreover, we prove that if we sample the corners and their neighbors we have \(C_{er} = 0\). We start by rewriting Equation 2.18 in a more convenient form. Using Lemma 30, we know that \(N(D_J)^T = [D^T]_{\mathcal{M},J}\). In words, \(D_J\) selects the rows of \(D\) at indices \(J\), or equivalently the columns of \(D^T\). Similarly the multiplication by \(N\) selects the rows of \(D^T\) at indices \(\mathcal{M}\). Similarly, \(N(D_I)^T = [D^T]_{\mathcal{M},I}\). Using these relations, Equation 2.18 simplifies to:

\[C_{er} = \|([D^T]_{\mathcal{M},J})^T[D^T]_{\mathcal{M},I}\|_{\infty \rightarrow \infty} \]  

(A.4)

Since \(I \cup J = \{1, \ldots, n\}\), it is clear that \([D^T]_{\mathcal{M},J}\) and \([D^T]_{\mathcal{M},I}\) are disjoint sets of columns of the matrix \([D^T]_{\mathcal{M}}\).

Let us start with the first claim: \(C_{er} = 1\) whenever we sample the corners, the first, and the last entry of a signal. We will make an extensive use of the structure of the matrix \(D^T\) which is the transpose of Equation 2.6. To give a more intuitive understanding of the proof
we provide a small example of $D^T$ with $n = 12$:

\[
\begin{array}{cccccccccccc}
-2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\
\end{array}
\]

(A.5)

The matrix $[D^T]_{\mathcal{M},\mathcal{I}}$ is obtained from $D^T$ after removing the rows at indices in the sample set $\mathcal{M}$: these “deleted” rows are shown in gray in Equation A.5. In particular, according to the assumptions of the first claim of Proposition 7, $\mathcal{M}$ contains the first and the last sample (first and last row in $D^T$) plus intermediate rows corresponding to corners (two intermediate gray rows in the figure). Now we note that the matrix $[D^T]_{\mathcal{M},\mathcal{I}}$ selects the columns with indices in $\mathcal{I}$ from $[D^T]_{\mathcal{M},\mathcal{I}}$. In figure, the columns that form $[D^T]_{\mathcal{M},\mathcal{I}}$ are shown in dashed red boxes. The position of these columns is dictated by the position of the corners, hence if the $i$-th row corresponds to a corner, than column $i-1$ belongs to $\mathcal{I}$.

Three considerations are in order now. First, the matrix $[D^T]_{\mathcal{M},\mathcal{I}}$ is a block-diagonal square matrix with diagonal blocks being Toeplitz matrices (cf. with Equation A.1). Second, the matrix is invertible (follows from the first claim of Lemma 31). Third, the matrix $[D^T]_{\mathcal{M},\mathcal{I}}$ only contains 0 and 1 in suitable positions. Therefore, the matrix $([D^T]_{\mathcal{M},\mathcal{I}})^+[D^T]_{\mathcal{M},\mathcal{I}} = ([D^T]_{\mathcal{M},\mathcal{I}})^{-1}[D^T]_{\mathcal{M},\mathcal{I}}$ has the following block structure:

\[
\begin{bmatrix}
T_1^{-1} \\
\vdots \\
T_K^{-1}
\end{bmatrix}
\begin{bmatrix}
0 & 0 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
1 & 0 & \ldots & 0 & 0 \\
1 & 0 & \ldots & 0 & 0 \\
0 & 1 & \ldots & 0 & 0 \\
\vdots
\end{bmatrix}
\begin{bmatrix}
T_1^{-1}R_1 \\
T_2^{-1}R_2 \\
\vdots \\
T_K^{-1}R_K
\end{bmatrix}
\]

(A.6)

where $T_1, \ldots, T_K$ are Toeplitz matrices of suitable dimensions and each $R_i$ contains at most two nonzero elements (equal to 1) in the first and the last row. Since for a matrix $M$, $\|M\|_{\infty\rightarrow\infty}$ is the maximum of the $\ell_1$-norm of each row, we only need to demonstrate that the maximum $\ell_1$-norm of the rows of $T_i^{-1}R_i$ is no larger than 1 for all $i$. The action of $R_i$ on $T_i^{-1}$ is to select the first and/or the last column of $T_i^{-1}$ (depending on where the 1 appears). For instance, $T_1^{-1}R_1$ is zero everywhere, except the fist column which is equal to the
last column of $T_i^{-1}$. From Lemma 31(ii) we know that the entries in the first column have magnitude smaller than 1 hence it follows that $\|T_1^{-1}R_1\|_{\infty\to\infty} < 1$. A similar argument holds for the last column, hence $\|T_K^{-1}R_K\|_{\infty\to\infty} < 1$. For the intermediate blocks $T_i^{-1}R_i$, $1 < i < K$, it can be seen that $\|T_i^{-1}R_i\|_{\infty\to\infty} = \|T_i^{-1}v\|_{\infty}$, with $v \doteq [1 \ 0 \ \ldots \ \ 0 \ 1]^T$; this follows from the fact that $R_i$ selects the first and the last columns of $T_i^{-1}$ which have negative entries due to Lemma 31(ii). Using Lemma 31(iii) we know that $T^{-1}v = -1$, from which it follows $\|T_i^{-1}R_i\|_{\infty\to\infty} = 1$. This proves the first claim.

The proof of the second claim ($C_{er} = 0$ when we sample the corners and their neighbors) is much simpler. Sampling the neighbors corresponds to deleting the rows contiguous to each “corner” from $D^T$. From Equation A.5 the reader can easily see that this choice makes $[D^T\mathcal{M},\mathcal{I}] = 0$, which in turns implies $C_{er} = \|(D^T\mathcal{M},\mathcal{I})^+[D^T\mathcal{M},\mathcal{I}]\|_{\infty\to\infty} = 0$. \hfill \square

### A.3 Proof of Proposition 8

In this appendix we prove that in 3D depth reconstruction problems, if we sample the edges and the corresponding vertical and horizontal neighbors, then:

$$C_{er} \doteq \|(N(\Delta_J)^T)^+N(\Delta_I)^T)\|_{\infty\to\infty} = 0 \quad (A.7)$$

which implies exact recovery of the original depth signal according to Proposition 6. As in Appendix A.2 we rewrite the condition (A.7) as:

$$C_{er} = \|([\Delta^T\mathcal{M},\mathcal{J}]^+[\Delta^T\mathcal{M},\mathcal{I}]\|_{\infty\to\infty} = 0 \quad (A.8)$$

The proof proceeds along the same line of the proof of the second claim in Proposition 7. By observing the structure of $\Delta^T$, we realize that sampling the edges and the corresponding vertical and horizontal neighbors, makes $[\Delta^T\mathcal{M},\mathcal{I}] = 0$, which in turns implies $C_{er} = 0$. \hfill \square

### A.4 Proof of Proposition 9

In this appendix we establish necessary and sufficient conditions for an estimate $z^*$ to be in the set $\mathcal{S}^*$ of optimal solutions of problem (2.9). The proof is identical for the 3D case in Corollary 10 (substituting $D$ with $\Delta$), hence we restrict ourselves to the 2D case. We rewrite (2.9) as:

$$\min_z \|Dz\|_1 + \chi\{Az=y\} \doteq \min_z f(z) \quad (A.9)$$

where $\chi\{Az=y\}$ is the indicator function of the set $\{z : Az = y\}$, which is zero whenever $Az = y$ and $+\infty$ otherwise. Since $Az = y$ defines a convex (affine) set, the problem (A.9) is convex. In the following we make extensive use of the notion of subgradients of convex functions. We refer the reader to [197, §4] for a comprehensive treatment and to [198] for a quick introduction.

A point $z^*$ is a minimizer of a convex function $f$ if and only if $f$ is subdifferentiable at $z^*$ and the zero vector belongs to the set of subgradients of $f$, i.e., $0 \in \partial f(z^*)$. The set of
subgradients is also called the subdifferential. The subdifferential of a sum of functions is the sum of the subdifferentials, therefore

$$
\partial f(z^*) = \partial(\|Dz\|_1)(z^*) + \partial(\chi_{\{A^Tz = y\}})(z^*)
$$

(A.10)

In the following we compute each subdifferential in (A.10). Let us call \(I\) the support set of the vector \(Dz^*\), and recall that, given a vector \(v\), we denote with \(v_I\) the subvector of \(v\) including the entries of \(v\) at indices in \(I\). Using [198, Page 5]:

$$
\partial(\|Dz\|_1)(z^*) = \{ D^Tu \in \mathbb{R}^n : u_I = \text{sign}(Dz^*)_I, \|u_J\|_{\infty} \leq 1 \}
$$

(A.11)

The second subdifferential in (A.10) is [199, Page 254]:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g^Tz^* \geq g^Tr, \ \forall r \text{ s.t. } Ar = y \}
$$

(A.12)

To get a better understanding of the set in (A.12), we note that every solution \(r \in \mathbb{R}^n\) of the overdetermined linear system \(Ar = y\) can be written as a vector that satisfies the linear system, plus a vector that is in the null space of \(A\). Now we know that \(z^*\), the vector that generated the data \(y\), satisfies \(Az^* = y\). Therefore, we rewrite (A.12) as:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g^Tz^* \geq g^T(z^* + \bar{r}), \ \forall \bar{r} \in \ker(A) \}
$$

(A.13)

where \(\ker(A)\) denotes the kernel of \(A\). From Lemma 30 we know that the kernel \(A\) is spanned by the matrix \(N\) (defined in the lemma), hence (A.13) further simplifies to:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g^T(z^* + Nw) \geq g^T(z^*), \ \forall w \in \mathbb{R}^m \}
$$

(A.14)

Rearranging the terms:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g^T(z^* - z^*) \geq (N^Tg)^T w, \ \forall w \in \mathbb{R}^m \}
$$

(A.15)

From the second claim of Lemma 30 we know that \(N^Tg = g_{\mathcal{M}}\); moreover, we observe that if an element of \(g_{\mathcal{M}}\) is different from zero, then we can pick an arbitrarily large \(w\) that falsifies the inequality, therefore, it must hold \(g_{\mathcal{M}} = 0\). Therefore, we rewrite (A.15) as:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g^T(z^* - z^*) \geq 0, \ g_{\mathcal{M}} = 0 \}
$$

(A.16)

Now we split the product \(g^T(z^* - z^*)\) as \(g^T_{\mathcal{M}}(z^* - z^*)_{\mathcal{M}} + g^T_{\mathcal{M}^c}(z^* - z^*)_{\mathcal{M}^c}\) and note that \(g_{\mathcal{M}} = 0\). Moreover, for any feasible \(z^*\), the \(i\)-th entry of \(z^* - z^*\) is zero for all \(i \in \mathcal{M}\), which implies \(g^T_{\mathcal{M}}(z^* - z^*)_{\mathcal{M}} = 0\). Therefore, the inequality \(g^T(z^* - z^*) \geq 0\) vanishes and we remain with:

$$
\partial(\chi_{\{A^Tz = y\}})(z^*) = \{ g \in \mathbb{R}^n : g_{\mathcal{M}} = 0 \}
$$

(A.17)

Substituting (A.17) and (A.11) back into (A.10), we obtain:

$$
\partial f(z^*) = \{ D^Tu + g : u_I = \text{sign}(Dz^*)_I, \|u_J\|_{\infty} \leq 1, g_{\mathcal{M}} = 0 \}
$$

(A.18)

We can now use the subdifferential (A.18) to describe the optimal solution set \(\mathcal{S}^*\) of (2.9); as
mentioned earlier in this section, \( z^* \) is optimal if and only if zero is a subgradient, therefore \( S^* \) is defined as:

\[
S^* = \{ z^* : \exists u \in \mathbb{R}^{n-2}, g \in \mathbb{R}^n, \text{ such that } D^T u + g = 0, \ u_I = \text{sign}(Dz^*)_I, \ ||u_J||_\infty \leq 1, \ g_M = 0 \} \tag{A.19}
\]

We note that the constraints \( D^T u + g = 0 \) and \( g_M = 0 \) can be written compactly as \( [D^T u]_M = 0 \), which is the same as \( (D^T)_M u = 0 \). This allows rewriting (A.19) as:

\[
S^* = \{ z^* : \exists u \in \mathbb{R}^{n-2}, \text{ such that } (D^T)_M u = 0, \ u_I = \text{sign}(Dz^*)_I, \ ||u_J||_\infty \leq 1 \} \tag{A.20}
\]

which coincides with the optimality condition of Proposition 9, proving the claim.

\[ \square \]

### A.5 Proof of Theorem 13

Theorem 13 says that sign consistency of \( z \) is a necessary and sufficient condition for \( z \) to be in the solution set of (2.9). In the following we denote with \( SC \) the set of sign consistent signals which are feasible for (2.9). Moreover, we denote with \( S^* \) the set of optimal solutions of (2.9). The proof relies on the optimality conditions of Proposition 9, which we recall here: a signal \( z \) is in the solution set \( S^* \) if and only if there exists a \( u \in \mathbb{R}^{n-2} \) such that

\[
(D^T)_M u = 0 \text{ and } u_I = \text{sign}(Dz)_I \text{ and } ||u_J||_\infty \leq 1 \tag{A.21}
\]

where \( I \) is the support set of the vector \( Dz \). Before proving that \( z \in SC \Leftrightarrow z \in S^* \), we need a better understanding of the structure of the matrix \( (D^T)_M \). We note that, when taking twin samples, the matrix \( (D^T)_M \) is obtained from \( D^T \) by removing pairs of consecutive rows. For instance, considering a problem with \( n = 12 \), the product \( (D^T)_M u \) becomes:

![Matrix Diagram](image)

where gray rows are the ones we “removed” from \( D^T \) to obtain \( (D^T)_M \). By observing (A.22), the reader can verify that the resulting matrix \( (D^T)_M \) is block diagonal (in general will have more than 2 diagonal blocks), and each block is a 2nd-order difference operator like (2.6) of suitable size. We denote the diagonal blocks as \( D^{(1)}, D^{(2)}, \ldots, D^{(K)} \) \( (K = 2 \) in the example of eq. (A.22)). This also induces a partition in the vector \( u \), which can be split vertically as \( u = [u_{S_1} \ u_{S_2} \ \ldots \ u_{S_K}] \). Geometrically, the block diagonal structure that arises means that...
each segment between consecutive twin samples can be studied independently. Therefore, the condition (A.21) can be written as:

\[ D^{(k)} u_{S_k} = 0 \text{ and } u_{I_k} = \text{sign}(Dz)_{I_k} \text{ and } \|u_{S_k}\|_\infty \leq 1 \]  \hspace{1cm} (A.23)

for \( k = 1, \ldots, K \), where \( I_k \subseteq S_k \) are entries in the support set \( I \) that fall within the set \( S_k \).

With this machinery we are ready to prove Theorem 13.

Let us start with the implication \( z \in SC \Rightarrow z \in S^* \). We first consider the case in which all signs are consistent, i.e., Definition 11(i). This means that within each set \( S_k \), \( \text{sign}(Dz)_{I_k} = +1 \) or \( \text{sign}(Dz)_{I_k} = -1 \). Without loss of generality, assume \( \text{sign}(Dz)_{I_k} = +1 \). Then, we can see that selecting \( u_{S_k} = +1 \) satisfies \( u_{I_k} = \text{sign}(Dz)_{I_k} \) and \( \|u_{S_k}\|_\infty \leq 1 \). Moreover, since \( 1 \) is in the null space of \( D^{(k)} \) (see Lemma 32), it follows \( D^{(k)} u_{S_k} = 0 \), proving the claim. To complete the demonstration of \( z \in SC \Rightarrow z \in S^* \) we consider the case in which there is a sign change at the boundary of each segment, while all signs are zero in the interior (Definition 11(ii)). In this case the condition \( u_{I_k} = \text{sign}(Dz)_{I_k} \) imposes that the first and the last elements of \( u_{I_k} \) are +1 and -1 (or -1 and +1) respectively. Without loss of generality, assume that the signs are +1 and -1. Then the linear system \( D^{(k)} u_{S_k} = 0 \) becomes \( T_k \hat{u}_{S_k} = \pm[+1 \ 0 \ldots \ 0 \ -1]^T \), where \( T_k \) is a Toeplitz matrix of suitable dimension and \( \hat{u}_{S_k} \) is the vector \( u_{S_k} \) without the first and the last entry which we fixed to +1 and -1, respectively. The existence of a suitable solution to the linear system \( T_k \hat{u}_{S_k} = \pm[+1 \ 0 \ldots \ 0 \ -1]^T \), which is such that \( \|T_k \hat{u}_{S_k}\|_\infty \leq 1 \) follows from Lemma 31(ii).

Let us prove the reverse implication, i.e., \( z \in S^* \Rightarrow z \in SC \). Without loss of generality, we consider a single segment \( S_k \) and we re-label the corresponding entries from 1 to \( n_k \). Let us assume that \( z \in S^* \), which means that there exists \( u_{S_k} \) such that \( D^{(k)} u_{S_k} = 0 \), \( u_{I_k} = \text{sign}(Dz)_{I_k} \), and \( \|u_{S_k}\|_\infty \leq 1 \). Any solution of \( D^{(k)} u_{S_k} = 0 \) is in the null space of \( D^{(k)} \), which is spanned by the vectors \( v_1 \) and \( v_2 \) defined in Lemma 32. Therefore, we write \( u_{S_k} \) as \( u_{S_k} = \alpha v_1 + \beta v_2 \) with \( \alpha, \beta \in \mathbb{R} \). Assume that there are indices \( i, j \in S_k \), such that \( \text{sign}(Dz)_i = +1 \) and \( \text{sign}(Dz)_j = -1 \), therefore it must hold:

\[ [\alpha v_1 + \beta v_2]_i = +1 \text{ and } [\alpha v_1 + \beta v_2]_j = -1 \]  \hspace{1cm} (A.24)

which, recalling the definitions of \( v_1 \) and \( v_2 \) in Lemma 32, becomes: \( \alpha + \beta i = +1 \) and \( \alpha + \beta j = -1 \). It follows that:

\[ \beta = \frac{2}{j - i} \]  \hspace{1cm} (A.25)

Now, since \( z \in S^* \) it must also hold \( \|u_{S_k}\|_\infty \leq 1 \) which can be written as:

\[ -1 \leq \alpha v_1 + \beta v_2 \leq 1 \Leftrightarrow 1 \leq \alpha + \beta h \leq 1, \forall h = 1, \ldots, n_k \]  \hspace{1cm} (A.26)

(A.27)

Combining the inequalities for \( h = 1 \) and \( h = n_k \) we get:

\[ \beta(n_k - 1) \leq 2 \]  \hspace{1cm} (A.28)

117
Substituting (A.25) into (A.28), we get:

\[
\frac{n_k - 1}{j - i} \leq 1
\]  

(A.29)

which is satisfied if and only if \( i = 1 \) and \( j = n_k \). Hence, we proved that sign changes of the curvature of \( z \in S^* \) can only happen at the boundary of each segment, which agrees with our definition of \( SC \) (Definition 11), proving the claim.

\[\square\]

### A.6 Proof of Proposition 14

Let us start by proving that if we sample the boundary of \( z^\circ \) and the sample set includes a twin sample in each linear segment of \( z^\circ \), then, \( z^\circ \) is in the set of minimizers of (2.9). The claim is a direct consequence of Theorem 13. By construction, if we have a twin sample per segment, only two cases are possible: either both samples fall inside the linear segment (i.e., none of them corresponds to a corner), or one of the samples corresponds to a corner. In both cases it is easy to see that \( z^\circ \) is sign consistent with respect to this choice of samples, which implies that \( z^\circ \in S^* \) according to Theorem 13.

The second claim states that any optimal solution \( z^* \) lies between the naive estimate \( \tilde{z} \) and the true signal \( z^\circ \). Before proving this claim we note that the presence of twin samples makes the objective of (2.9) separable. To see this we note that, in a noiseless case, if we sample a point \( i \), then its value is fixed by the corresponding linear constraint and \( z_i \) is no longer a variable. Therefore, by sampling, we are essentially fixing pairs of consecutive entries in \( z \). Using this property, we see that the objective separates as:

\[
\|Dz\|_1 = \|D^{(1)}z_{S_1}\|_1 + \|D^{(2)}z_{S_2}\|_1 + \ldots + \|D^{(K)}z_{S_K}\|_1,
\]  

(A.30)

where \( D^{(k)} \) is a 2nd-order difference matrix of suitable dimensions, and \( z_{S_k} \) is the subvector of \( z \) including the entries corresponding to consecutive twin samples (say \( z_{i-1}, z_i \) and \( z_j, z_{j+1} \), which are fixed to known values) and all the entries between those (i.e., \( z_{i+1}, \ldots, z_{j-1} \)); we used the symbol \( K \) in (A.30) to denote the number of regions between consecutive twin samples. From the separability of the objective, it follows that we can study each region (between twin samples) independently (the optimization splits in \( K \) independent optimizations). We now prove the second claim: for any optimal solution \( z^* \in S^* \) and any index \( i \in \{1, \ldots, n\} \), it holds that \( \min(z_i^\circ, \tilde{z}_i) \leq z_i^* \leq \max(z_i^\circ, \tilde{z}_i) \). As mentioned before, this means that any optimal solution is “between” the naive solution \( \tilde{z} \) (obtained by connecting the dots, see the blue dashed line in Figure 2-5(a)) and the true solution \( z^\circ \) (black solid line in Figure 2-5(a)). We show that the claim must hold true in all regions \( S_1, \ldots, S_K \). First, let us get rid of the “degenerate” regions: these are the ones in which \( \tilde{z}_i = z_i^\circ \) for all \( i \in S_k \). This happens when we sample a corner and there are 3 collinear samples as in Figure A-1. In this case for any index \( i \in S_k \), we prove \( \tilde{z}_i = z_i^* = z_i^\circ \), i.e., all optimal solutions must reduce to a straight line between the collinear samples. We prove this with the visual support of Figure A-1. Our goal is to show that any \( \tilde{z} \) that deviates from linearity is not sign consistent (SC), hence cannot be optimal. If a sample \( k \in \{i + 1, \ldots, j - 1\} \) has a curvature different from zero, then, to be sign consistent it...
cannot change curvature. This case is shown with the label (↓) in Figure A-1. Clearly, if \( \hat{z} \) cannot change curvature, after deviating from the straight line, it cannot reach the sample \( j \), leading to contradiction. Similarly, contradiction occurs when the curvature is different from zero at \( i \): a positive curvature at \( i \) (case (↑↓) in figure), must be compensated by a negative curvature before \( j \) for the curve to intersect \( j \): this again violates sign consistency; analogous argument holds for a negative curvature at \( i \) (case (↓↑) in figure). Therefore we proved that in these straight segments it holds \( \tilde{z}_i = z_i^{\dag} = z_i^\diamond \). If only remains to discuss the case in which \( \tilde{z}_i \neq z_i^\diamond \) which occurs whenever consecutive double samples do not include corners. This situation is pictured in Figure 2-5(a); in this case \( \tilde{z}_i\leq z_i^\diamond \). We only prove that whenever \( \tilde{z}_i \leq z_i^\diamond \) then \( \tilde{z}_i \leq z_i^\dag \leq z_i^\diamond \); the proof for the case \( z_i^\diamond \leq \tilde{z}_i \) is practically identical (the corner points downwards). In Figure 2-5(a) we show two cases (dashed lines): in case (↑↓) we show a signal \( \hat{z} \) above \( z^\diamond \); in (↓↑) we show a signal below \( \tilde{z} \). One can easily realize than any signal as in case (↑↓) has a positive curvature at sample \( i \) and a negative curvature at the top corner: this violates sign consistency hence \( \hat{z} \) is not in the solution set. A similar argument holds in the case (↓↑), which concludes the proof that \( z^* \) must be “between” \( \tilde{z} \) and \( z^\diamond \).

We conclude the proof by deriving the error bound in eq. (2.20), repeated below for the reader’s convenience:

\[
\|z^\diamond - z^*\|_\infty \leq \max_{i\in\mathcal{M}} d_i \cos(\theta_i) \tag{A.31}
\]

where \( d_i \) is the distance between the sample \( i \) and the nearest corner in \( z^\diamond \), while \( \theta_i \) is the angle that the line connecting \( i \) with the nearest corner forms with the vertical, see Figure 2-5(a). The bound relates any solution \( z^* \) of the \( \ell_1 \)-minimization problem (2.9) to the true signal \( z^\diamond \).

To prove (A.31), we note that \( z^* \) can deviate from \( z^\diamond \) only in the “corner cases” as the one in Figure 2-5(a) (proven above in this section). Moreover, we note that the maximum error \( |z_i^\diamond - \tilde{z}_i| \) is attained at the corner of \( z^\diamond \) and is denoted with \( \hat{d} \) in the figure. From basic trigonometry we conclude:

\[
\hat{d} \leq \max_{k\in\{i,j\}} d_k \cos \theta_k \tag{A.32}
\]
(in Figure 2-5(a) this becomes: \( \hat{d} \leq d_i \cos(\theta_i) \)), where \( d_k \) is the distance between sample \( k \) and the nearest corner, while \( \theta_k \) is the angle that the line connecting sample \( k \) with the nearest corner forms with the vertical. The bound (A.31) follows by extending this inequality to all linear segments in \( z^\circ \).

\[ (\Delta^T)_{\mathcal{M}_i} = \begin{bmatrix} 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 \end{bmatrix} \]

\[ (\Delta^T)_{\mathcal{M}_i} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \]

Figure A-2: Example of matrix \((\Delta^T)_{\mathcal{M}}\) and corresponding column partition \((\Delta^T)_{\mathcal{M},V}\), \((\Delta^T)_{\mathcal{M},H}\).

A.7 Proof of Theorem 17

In this section we prove that if a 3D signal \( Z \) is feasible for problem (2.15) and it is 3D sign consistent, then it is also a minimizer of (2.15). The proof is similar to the 2D case in Proposition 14, and relies on Corollary 10, which we restate as follows: given a signal \( Z \in \mathbb{R}^{r \times c} \) which is feasible for problem (2.15), \( Z \) is in the optimal set of (2.15) if there exists a vector \( u \in \mathbb{R}^{2(n-r-c)} \), with \( n = r \times c \), such that

\[ (\Delta^T)_{\mathcal{M}} \ u = 0 \quad \text{and} \quad u_I = \text{sign}(\Delta z)_I \quad \text{and} \quad \|u_J\|_\infty \leq 1, \quad (A.33) \]

where \( \mathcal{M} \) denotes a grid sample set. Let \( \mathcal{M}_i \) be patches defined in Definition 15. We show that 3D sign consistency of \( Z \) with respect to grid samples \( \mathcal{M} \) implies (A.33). We start by noting that when using grid samples with \( K \) patches (see Figure 2-6(a)), problem (2.15) separates into \( K \) independent optimization subproblems (similarly to the 2D case of eq. (A.30)). Therefore, without loss of generality in the following we focus on a single patch and we assume that the grid samples include the boundaries of the patch (first and last two rows and columns). With slight abuse of notation we denote this patch with \( Z \) and we use \( z = \text{vec}(Z) \).

Before proving the claim we need some insight on the structure of the matrix \((\Delta^T)_{\mathcal{M}}\). This matrix is obtained by deleting rows of \( \Delta^T \) indexed by \( \mathcal{M} \). Since we are assuming to sample the boundaries of the patch, the resulting \((\Delta^T)_{\mathcal{M}}\) has the structure described in Figure A-2. Now we remain to show that when \( Z \) is 3D sign consistent, we can find a vector \( u \) that satisfies the three conditions in (A.33). Towards this goal, we recall that \( \Delta \) is obtained by stacking two submatrices that compute the vertical and horizontal differences
as in (2.16). Therefore, we split $\text{sign}(\Delta z)_I$ accordingly as:

$$
\text{sign}(\Delta z)_I = \begin{bmatrix}
\text{sign}(\Delta z)_{I_V} \\
\text{sign}(\Delta z)_{I_H}
\end{bmatrix}
$$

(A.34)

where $I_V$ includes indices in the support set $I$ corresponding to nonzero vertical differences, while $I_H$ includes indices in $I$ corresponding to nonzero horizontal differences.

Now, we note that the row partition of $\Delta$ induces a column partition of $(\Delta^T)\mathcal{M}$. We call the corresponding submatrices $(\Delta^T)\mathcal{M},V$ and $(\Delta^T)\mathcal{M},H$, as shown in Figure A-2. This also partitions the vector $u$ into two subvectors $u_V$ and $u_H$. Therefore, we rewrite the condition $(\Delta^T)\mathcal{M}u = 0$ as:

$$
(\Delta^T)\mathcal{M},V u_V + (\Delta^T)\mathcal{M},H u_H = 0
$$

(A.35)

Since $Z$ is 3D sign consistent, then $\text{sign}(\Delta z)_{I_V}$ is either $+1$ or $-1$ and $\text{sign}(\Delta z)_{I_H}$ is either $+1$ or $-1$. Assume without loss of generality that $\text{sign}(\Delta z)_{I_V} = +1$ and $\text{sign}(\Delta z)_{I_H} = -1$. Now if we choose $u_V = +1$ and $u_H = -1$, it holds that $u_{I_V} = \text{sign}(\Delta z)_{I_V}$, $u_{I_H} = \text{sign}(\Delta z)_{I_H}$, and $\|u_J\|_\infty \leq \|u\|_\infty \leq 1$, hence the last two conditions in (A.33) are satisfied. Moreover, since each row of $(\Delta^T)\mathcal{M},V$ and $(\Delta^T)\mathcal{M},H$ includes only three nonzero entries with values $+1$, $-2$, $+1$, it follows that:

$$
(\Delta^T)\mathcal{M},V u_V = (\Delta^T)\mathcal{M},V 1 = 0 \\
(\Delta^T)\mathcal{M},H u_H = -(\Delta^T)\mathcal{M},H 1 = 0
$$

(A.36)

which implies (A.35), concluding the proof.

\section*{A.8 Proof of Proposition 18}

By the assumptions of Proposition 18, $Z^\circ$ is the ground truth generating noiseless measurements (2.4) and $Z^\circ$ is 3D sign consistent with respect to $\mathcal{M}$. Then each row of $Z^\circ$, namely $Z^\circ_i$, is a sign consistent 2D depth signal and, given the samples, we can build a row-wise envelope for $Z^\circ_i$ as prescribed in Theorem 13 (pictorial explanation in Figure 2-6(b)). Repeating this procedure for all rows $i$ and calling $\bar{Z}$ and $\underline{Z}$ the point-wise upper bound an lower bound for the envelope of each row, then $Z^\circ$ is within the row-wise envelope, i.e., $Z_{i,j} \leq Z^\circ_{i,j} \leq \bar{Z}_{i,j}$ for all $i = 1, \ldots, r$ and $j = 1, \ldots, c$. Therefore, given an optimal solution $Z^\star$ of (2.15) it holds that

$$
|Z_{i,j}^\circ - Z_{i,j}^\star| \leq \max(|Z_{i,j} - Z_{i,j}^\star|, |\bar{Z}_{i,j} - Z_{i,j}^\star|).
$$

which trivially follows from the chain of inequalities $Z_{i,j} \leq Z_{i,j}^\circ \leq \bar{Z}_{i,j}$, concluding the proof.

\section*{A.9 Proof of Proposition 19}

This section proves the necessary and sufficient conditions for an estimate $z^\star$ to be in the set $\mathcal{S}^\star$ of optimal solutions of problem (2.10), as stated in Proposition 19. We follow similar
arguments as the proof in Appendix A.4.

We start by rewriting Problem 2.10 as:

\[
\min_z \|Dz\|_1 + \chi_{\{z: \|Az-y\|_\infty \leq \varepsilon\}} = \min_z f(z) \quad (A.37)
\]

As discussed earlier in Appendix A.4, a signal \(z^*\) is optimal for problem (2.10) if the zero vector belongs to the set of subgradients of \(f\) at \(z^*\), i.e., \(0 \in \partial f(z^*)\). The subdifferential of \(\|Dz\|_1\) was given in (A.11). The subdifferential of the indicator function in (A.37) is given, similarly to (A.12), as

\[
\partial(\chi_{\{z: \|Az-y\|_\infty \leq \varepsilon\}})(z^*) = \{g \in \mathbb{R}^n : g^Tz^* \geq g^Tr, \ \forall r \text{ s.t. } \|Ar-y\|_\infty \leq \varepsilon\} \quad (A.38)
\]

Recall that the matrix \(A\) restricts a vector \(r\) to its entries in the sample set \(\mathcal{M}\), therefore we have

\[
\{r : \|Ar-y\|_\infty \leq \varepsilon\} = \{r : \|r_\mathcal{M}-y\|_\infty \leq \varepsilon\}. \quad (A.39)
\]

Obviously \(\|r_\mathcal{M}-y\|_\infty \leq \varepsilon\) implies for all \(i \in \mathcal{M}\) that

\[
r_i \leq y_i + \varepsilon \quad \text{or} \quad r_i \geq y_i - \varepsilon. \quad (A.40)
\]

We decompose

\[
g^Tz^* \geq g^Tr = g^T_{\mathcal{M}}r_\mathcal{M} + g^T_{\mathcal{M}^c}r_{\mathcal{M}^c}. \quad (A.41)
\]

A vector \(g\) that satisfies (A.41), for all \(r\) obeying (A.39), must satisfy \(g^T_{\mathcal{M}^c} = 0\) since \(r_{\mathcal{M}^c}\) is a free variable. With this at hand, we proceed as

\[
g^Tz^* \geq \max_{\|r_\mathcal{M}-y\|_\infty \leq \varepsilon} g^T_{\mathcal{M}}r_\mathcal{M} = \max_{\|r_\mathcal{M}-y\|_\infty \leq \varepsilon} \sum_{i \in \mathcal{M}} g_ir_i
\]

\[
= \max_{\|r_\mathcal{M}-y\|_\infty \leq \varepsilon} \left( \sum_{g_i \geq 0, i \in \mathcal{M}} |g_i|y_i - \sum_{g_i < 0, i \in \mathcal{M}} |g_i|y_i \right)
\]

\[
= \sum_{g_i \geq 0, i \in \mathcal{M}} |g_i|(y_i + \varepsilon) - \sum_{g_i < 0, i \in \mathcal{M}} |g_i|(y_i - \varepsilon)
\]

\[
= \sum_{g_i \geq 0, i \in \mathcal{M}} |g_i|y_i - \sum_{g_i < 0, i \in \mathcal{M}} |g_i|y_i + \varepsilon \sum_{i \in \mathcal{M}} |g_i|
\]

\[
= g^T_{\mathcal{M}}y + \varepsilon \|g_\mathcal{M}\|_1.
\]

Rearranging terms yields

\[
\|g_\mathcal{M}\|_1 \leq \frac{g^T_{\mathcal{M}}(z^*_\mathcal{M} - y)}{\varepsilon}. \quad (A.42)
\]
We can then write the solution set as

\[ S^* = \{ z^* : \exists u \in \mathbb{R}^{n-2}, g \in \mathbb{R}^n, \text{ such that } D^T u + g = 0, \ u_I = \text{sign}(Dz^*)_I, \]

\[ \|u_I\|_\infty \leq 1, \ \|g_M\|_1 \leq g^T_M(z^*_M - y)/\varepsilon, \ g^T_M = 0 \}. \tag{A.43} \]

The conditions \( D^T u + g = 0 \) and \( g^T_M = 0 \) can be cast as \( (D^T)_M u = 0 \) and \( g_M = -(D^T)_M u \). Then inequality in (A.42) must be an equality since \( |z^*_M - y| \leq \varepsilon \) (hence \( \|z^*_M - y\|_\infty \leq 1 \)) and \( \|g_M\|_1 = \max_{\|v\|_\infty \leq 1} g^T_M v \) by definition of the \( \ell_1 \)-norm; moreover if \( |z^*_i - y_i| = \varepsilon \) for some \( i \in \mathcal{M} \) it must hold that

\[ \text{sign}((D^T)_i u) = (z^*_i - y_i)/\varepsilon. \tag{A.44} \]

Here we used the assumption that \( \varepsilon > 0 \). Denote the set of such \( i \in \mathcal{M} \) as the active set \( \mathcal{A} \) as in (2.22). For \( i \in \mathcal{M} \setminus \mathcal{A} \), it holds that \( (D^T)_i u = 0 \). Splitting \( \mathcal{A} \) into two subsets as in (2.23) and using (A.44) yields the conditions (2.25) of Proposition 19.

\[ \square \]

### A.10 Proof of Proposition 23

To prove the claim of Proposition 23, we show that any 2D sign consistent signal is bounded above by \( \bar{z} \) and below by \( \underline{z} \) as defined in Definition 22. We restrict the proof to demonstrate the validity of the upper bound, since the argument for the lower bound follows similarly.

To this end, we focus on line segments (1), (3) and (5) that determine \( \bar{z} \). Figure A-3(a)-(d) illustrate 4 possible orientations of these line segments and the resulting upper bounds (in solid blue line).

Let us start with the case of Figure A-3(a). We use a contradiction argument similarly to the proof of Proposition 14. We show that a signal which is not upper bounded by the solid blue line, that is \( \max\{(1), (3)\} \) in this case, has to be sign inconsistent (Definition 11). Indeed such a signal needs to have a nonzero curvature at sample either \( i + 1 \) or \( j \) since the line segments (1) and (3) represent the extreme slopes that a signal can have with zero curvature at the end points. Assume without loss of generality, a signal (dashed black line in Figure A-3(a)) that has positive curvature at \( i + 1 \) (label \( \downarrow \uparrow \) in figure) and violates the upper bound. It is clear that this signal cannot reach the \( \varepsilon \)-interval (red bar in figure) at sample \( j \) without having a negative curvature between \( i + 1 \) and \( j \); this violates sign consistency (case (i) in Definition 11), leading to contradiction.

Next we prove the claim for Figure A-3(b). In this case the upper bound \( \bar{z} \) is given by the line segment (5). Any signal that is not upper-bounded by (5) needs to have a positive curvature (\( \downarrow \uparrow \)) between \( i + 1 \) and \( j \). It is obvious that such a signal also needs to have a negative curvature at some other sample \( k \in (i + 1, j) \) to reach the \( \varepsilon \)-interval at sample \( j \), which leads to contradiction.

Finally, observe that Figure A-3(c) and Figure A-3(d) are exactly symmetrical cases with different orientations for line segments (1) and (3). Therefore we consider only Figure A-3(c) here. Similarly to before, a signal (dashed black line in figure) that is not upper bounded by (5) needs to have a positive curvature between \( i + 1 \) and \( j \). However in order to reach the \( \varepsilon \)-bar at sample \( i + 1 \), this signal must have a negative curvature at some other
sample \( k \in (i + 1, j) \), which contradicts sign consistency.

We proved the claim of the proposition for all possible cases in Figure A-3 which ends our proof.
Figure A-3: Illustration of the upper bound $\bar{z}$ of 2D sign consistent $\epsilon$-envelope for all possible orientations of the line segments (1), (3) and (5) defined in Definition 22.
A.11 Proof of Proposition 24

From Theorem 21 we learn that any optimal solution \( z^\star \) must be 2D sign consistent, implying that it lies within the 2D sign consistent \( \epsilon \)-envelope. By the assumptions of Proposition 24 that we sample the boundary of \( z^\circ \) and the sample set includes a twin sample in each linear segment in \( z^\circ \), it follows that the ground truth signal \( z^\circ \) is also sign consistent (see discussion in Appendix A.6), hence belongs to the 2D sign consistent \( \epsilon \)-envelope as a result of Proposition 23.

Since both \( z^\star \) and \( z^\circ \) lie inside the same 2D envelope, the difference between \( z^\star_k \) and \( z^\circ_k \) for arbitrary \( k \in (i + 1, j) \) is bounded by the difference between the top and the bottom of the envelope, \( \bar{z}_k - \underline{z}_k \).

A.12 Proof of Corollary 26

In the proof of Proposition 14 (Appendix A.6) we have seen that the objective of (2.9) is separable as in (A.30), where \( S_k \) includes all indices falling in the region between two consecutive twin samples (e.g., if the twin samples are \( (i - 1, i) \) and \( (j, j + 1) \), then \( S_k = \{i - 1, i, i + 1, \ldots, j - 1, j, j + 1\} \)). This allows us to study the performance of Algorithm 1 independently for each region \( S_k \). As a result of Proposition 14, line 2 in Algorithm 1 produces a solution \( z^\star \) between \( \tilde{z} \) and \( z^\circ \). Assume without loss of generality that \( z^\star \) is concave in the region \( S_k \), e.g., the leftmost corner of Figure 2-2(b), which yields \( s_k = -1 \) in line 5. Then \( \tilde{z}_i \leq z^\star_i \leq z^\circ_i \) for all \( i \in S_k \) which implies

\[
\sum_{i \in S_k} z^\star_i \leq \sum_{i \in S_k} z^\circ_i \quad (A.45)
\]

Now we note that maximizing \( \sum_{i \in S_k} z_i \) is the same as minimizing \( -\sum_{i \in S_k} z_i \). Therefore, when \( s_k = -1 \), line 8 maximizes the objective \( \sum_{i \in S_k} z_i \) subject to \( z \in S^\star \) where \( S^\star \) is the optimal set of (2.9). Since (A.45) holds for all \( z \in S^\star \), and since \( z^\circ \in S^\star \), Algorithm 1 returns \( z^\circ \).

A.13 Computation for NESTA

In this appendix we provide some technical details as well as closed-form expressions related to Algorithm 2.

The function \( f_\mu \) in (2.33) is shown to be differentiable by Nesterov [160] and has gradient

\[
\nabla f_\mu(z) = D^T u^\star(z) \quad (A.46)
\]

where \( u^\star(z) \) is the optimal solution of the maximization in (2.33) and, for any given \( z \), can
be computed as

$$u^*(z) = \begin{cases} 
\mu^{-1}(Dz)_i, & \text{if } |(Dz)_i| < \mu \\
\text{sign}(Dz)_i, & \text{otherwise.}
\end{cases} \quad (A.47)$$

The gradient $\nabla f_\mu(z)$ is said to be Lipschitz with Lipschitz constant $L_\mu$ if it obeys

$$\|\nabla f_\mu(x) - \nabla f_\mu(z)\|_2 \leq L_\mu \|x - z\|_2.$$ 

The constant $L_\mu$, used in Algorithm 2, is shown to be $L_\mu = \frac{\|D\|_2}{\mu}$ in [138, eq (3.4)], where $\| \cdot \|$ denotes the spectral norm of a matrix.

Next we provide closed-form solutions for the optimization problems in lines 8-9 of Algorithm 2. We start with $\bar{q}$, which is the solution of the optimization problem in line 8. Eliminating constant terms in the objective in line 8 and completing the squares yields:

$$\bar{q} = \arg\min_{z \in Q} \frac{L_\mu}{2} \|z - z^{(k)}\|_2^2 + \langle g, z \rangle = \arg\min_{z \in Q} \|z - (z^{(k)} - L_\mu^{-1}g)\|_2^2 \quad (A.48)$$

where $g = \nabla f_\mu(z^{(k)})$ and $Q = \{z : \|Az - y\|_\infty \leq \varepsilon\}$. For simplicity we introduce the vector $\hat{z}^{(k)} = z^{(k)} - L_\mu^{-1}g$, and recall that the sampling matrix $A$ restricts a vector to its entries at indices in $M$, hence $Q = \{z : \|z_M - y\|_\infty \leq \varepsilon\}$. We then notice that the objective in (A.48) separates as:

$$\bar{q} = \arg\min_{z_M \in Q} \|z - \hat{z}^{(k)}\|_M^2 + \|z - \hat{z}^{(k)}\|_{\overline{M}}^2 \quad (A.49)$$

which further separates into two independent optimization problems involving subvectors of $\bar{q}$:

$$\bar{q}_M = \arg\min_{z_M \in Q} \|z - \hat{z}^{(k)}\|_M^2 \quad (A.50)$$

$$\bar{q}_{\overline{M}} = \arg\min_{z_{\overline{M}}} \|z - \hat{z}^{(k)}\|_{\overline{M}}^2 \quad (A.51)$$

Problem (A.51) is an unconstrained minimization, since $Q$ only constraints $z_M$. By inspection, problem (A.51) admits the trivial solution $\bar{q}_M = z^{(k)}_M$. It remains to solve (A.50):

$$\bar{q}_M = \arg\min_\zeta \|\zeta - \hat{z}^{(k)}\|_M^2 \quad \text{s.t. } \|\zeta - y\|_\infty \leq \varepsilon \quad (A.52)$$

where we “renamed” the optimization variable to $\zeta$ to simplify notation. Problem (A.52) is nothing but the projection of the vector $\hat{z}^{(k)}_M$ onto the $\ell_\infty$-ball with radius $\varepsilon$ centered at $y$. This projection can be explicitly calculated as:

$$\bar{q}_i = \min\{\max\{\hat{z}^{(k)}_i, y_i - \varepsilon\}, y_i + \varepsilon\}, \quad \forall i \in M.$$ 

which completes the derivation of the closed-form solution for $\bar{q}$. The reader may notice that the optimization problem in line 9 of Algorithm 2, whose solution is $\bar{w}$, is identical to the one in (A.48), after replacing $g = \sum_{i=0}^k \alpha_i \nabla f_\mu(z^{(i)})$ and using $z^{(0)}$ instead of $z^{(k)}$. For this reason the closed-form solution $\bar{w}$ can be derived in full similarity with the one of $\bar{q}$.
and can be explicitly written as:

\[ \vec{w}_M = \vec{z}^{(k)}_M \]

\[ \vec{w}_i = \min \{ \max \{ \vec{z}_i^{(k)}, y_i - \varepsilon \}, y_i + \varepsilon \}, \forall i \in M, \]

where \( \vec{z}^{(k)} \equiv z^{(0)} - L_\mu^{-1} \sum_{i=0}^{k} \alpha_i \nabla f_\mu(z^{(i)}) \).
Appendix B

Analysis and Proofs for Chapter 4

In this section we present the proofs for Chapter 4.

B.1 Notation

<table>
<thead>
<tr>
<th>Notation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbf{x}, \mathbf{z} )</td>
<td>two arbitrary, non-zero vectors</td>
</tr>
<tr>
<td>( \mathbf{x}^{(i)}, \mathbf{z}^{(i)} )</td>
<td>the corresponding output vectors at layer ( i \in {0, 1, 2} )</td>
</tr>
<tr>
<td>( \mathbf{x}_j^{(i)}, \mathbf{z}_j^{(i)} )</td>
<td>the ( j )th block of ( \mathbf{x}^{(i)}, \mathbf{z}^{(i)} ), no permutation</td>
</tr>
<tr>
<td>( \tilde{\mathbf{x}}^{(i)}, \tilde{\mathbf{z}}^{(i)} )</td>
<td>the permuted vectors at layer ( i \in {0, 1, 2} )</td>
</tr>
<tr>
<td>( \mathbf{x}_j^{(i)}, \mathbf{z}_j^{(i)} )</td>
<td>the ( j )th block of ( \mathbf{x}^{(i)}, \mathbf{z}^{(i)} ) after permutation</td>
</tr>
<tr>
<td>( D_0, D_1, D_2 )</td>
<td>number of blocks in a vector (before 1st layer, after permutation, and after 2nd layer, respectively)</td>
</tr>
<tr>
<td>( W_1, W_2 )</td>
<td>block weight matrices at each convolution layer</td>
</tr>
<tr>
<td>( W_{i,+}, \mathbf{z} )</td>
<td>effective weight matrices with ReLU taken into account, when input vector is ( \mathbf{z} )</td>
</tr>
<tr>
<td>( Q_{\mathbf{x}, \mathbf{z}} )</td>
<td>expectation of ( \mathbf{W}<em>{i,+}^{T} \mathbf{W}</em>{i,+} )</td>
</tr>
<tr>
<td>( \theta^{(i)} )</td>
<td>the angle between two different vectors ( \mathbf{x}, \mathbf{z} ) at layer ( i )</td>
</tr>
<tr>
<td>( \theta_j^{(i)} )</td>
<td>the angle between the ( j )th blocks of ( \mathbf{x}_j^{(i)}, \mathbf{z}_j^{(i)} ) at layer ( i ), without permutation</td>
</tr>
<tr>
<td>( \tilde{\theta}_j^{(i)} )</td>
<td>the angle between the ( j )th blocks of ( \tilde{\mathbf{x}}_j^{(i)}, \tilde{\mathbf{z}}_j^{(i)} ) at layer ( i ), with permutation</td>
</tr>
<tr>
<td>( h_{\mathbf{x}, \mathbf{z}} )</td>
<td>a random vector that ( W_{1,+}^{T} \mathbf{W}<em>{2,+}^{T} \mathbf{W}</em>{2,+} \mathbf{w}_{1,+} \mathbf{z} ) concentrates around</td>
</tr>
<tr>
<td>( h_{\mathbf{z}, \mathbf{z}^\circ} )</td>
<td>the perturbation of ( h_{\mathbf{z}, \mathbf{z}^\circ} ) around its mean ( \mathbf{z}/4 )</td>
</tr>
<tr>
<td>( S_{\mathbf{z}, \mathbf{z}^\circ} )</td>
<td>a small region outside which the perturbation ( h_{\mathbf{z}, \mathbf{z}^\circ} ) is very small</td>
</tr>
<tr>
<td>( v_{\mathbf{z}, \mathbf{z}^\circ} )</td>
<td>descent direction at ( \mathbf{z} )</td>
</tr>
</tbody>
</table>

Before continuing to the proof of Theorem 27, we introduce some notation.
Let $\text{Id}_n$ be an $n \times n$ identity matrix. If dimension is not specified, we assume it is clear from the context. Let $\text{diag}(Az > 0)$ be a diagonal matrix, where $(i, i)^{th}$ is 1 if $(Az)_i > 0$, and 0 otherwise. Let $B(z, r)$ be an Euclidean ball of radius $r$ centered at $z$. Let $W_{1,+,z} = \text{diag}(W_1z > 0)W_1$ and $W_{2,+,z} = \text{diag}(W_2W_1z > 0)W_2$. For matrices, $\|A\|$ denotes the spectral norm. Let $S^{k-1}$ be the unit sphere in $\mathbb{R}^k$. A block vector $z = [z_i]_i^n \in \mathbb{R}^{kn}$ is a concatenation of $n$ vectors, each of size $k$, and uses boldface notation. Similarly, a diagonal block matrix is denoted $W = [W_i]_i^n$, with matrices $\{W_i\}$ on it diagonal. For any nonzero $z \in \mathbb{R}^k$, let $\bar{x} = \frac{x}{\|x\|_2}$. For block vector $z = [z_i]_i^n$, let $\bar{z} = [\bar{z}_i]_i^n$. For fixed $x, z \in \mathbb{R}^k$, let $M_{x+z}$ be the matrix such that $M_{x+z}\bar{x} = \bar{z}$, $M_{x+z}\bar{z} = \bar{x}$ and $M_{x+z}v = 0$ for all $v \in \text{span} \{\{x, z\}\}^\perp$. Then given block vectors $x = [x_i]_i^n$, $z = [z_i]_i^n$, let $M_{x+z} = [M_{x_i+z_i}]_i^n$. Denote the block identity matrix $I = [\text{Id}_k]_i^n$. Let $\angle(x, z)$ be the angle between two vectors $x$ and $z$.

Recall that the weight matrix $W_i$ each for layers $i = 1, 2$ is assumed to be permuted to be a block matrix, as illustrated in Figure 4-2(b). A corresponding permutation is also applied to the input vectors of each layer.

Specifically, assume an input block vector $z = z^{(0)} = [z_j^{(0)}]_j^{D_0}$, then the output of the first transposed convolution layer is $z^{(1)} = W_{1,+}z^{(0)}$ which also has $D_0$ channels (blocks). Before the second convolution layer, we apply another permutation such that the new vector $\tilde{z}^{(1)} = \text{Perm}(z^{(1)})$ now has $D_1$ blocks. $\tilde{z}^{(1)}$ is then fed as an input to the second transposed convolution layer, resulting in an output $z^{(2)} = W_{2,+}\tilde{z}^{(1)}$.

In addition, let $\theta(i) = \angle(x^{(i)}, z^{(i)})$ denote the angle between two different vectors $x^{(i)}, z^{(i)}$ at the $i^{th}$ layer. In particular, let $\theta(j) = \angle(x^{(i)}, z^{(i)})$ denote the angle between the $j^{th}$ blocks of the two vectors. If the vectors are permuted, we use $\hat{\theta}(j) = \angle(x^{(i)}, z^{(i)})$. We also introduce the notation $\odot$ for multiplication of a regular vector $a \in \mathbb{R}^n$ and a block vector $z = [z_j]_j^n$ in the following way:

$$a \odot z = [a_z]_{j=1}^n.$$

We use big-$O(\cdot)$ notation to denote the order of magnitude for a variable. Finally, we also use $\lesssim, \gtrsim$ and $\asymp$ when the inequalities and equalities are up to a small universal constant $\epsilon$ which may not be specified. For instance $x \asymp y$ indicates that $x = y + O(\epsilon)$. All vector and angle notations used are summarized in Table B.1.

### B.2 Proof for Theorem 27

Proof mostly follows the arguments in [188]. As we discussed in Section 4.2, the weight matrix $W_1 \in \mathbb{R}^{C_1D_1 \times C_0D_0}$ of the first layer of the network can be arranged as a block matrix $W_1 = [W_i]_i^{D_0}$ where $W \in \mathbb{R}^{C_1 \times C_0}$ is a Gaussian matrix repeating in each block, see Figure 4-2. In the rest of the proof we will use this arrangement of the matrix. Note that this effectively means a permutation of the vectors after each layer. This has to be handled carefully throughout the proof. By assumption, the sampling matrix $A$ is an identity matrix, so the cost function can be written as

$$J(z) = \frac{1}{2}\|G(z^*) - G(z)\|^2_2.$$
The operation of the first layer on an input signal $z = [z_i]_{i=1}^{D_0}$ is

$$W_{1,+} z = \sigma(W_1 z).$$

In general operation of the generator network can be written as

$$G(z) = W_{2,+} W_{1,+} z.$$

**Remark 33.** The matrix $W_{1,+} z$ captures the operation of ReLU activation combined with weights of each layer, hence it will be instrumental in the rest of the proof. We note that that in the case of Leaky ReLU activation $L(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases}$, the output of each layer is

$$L(W_1 z) = \text{diag}(W_1 z > 0) W_1 z + \alpha \text{diag}(W_1 z < 0) W_1 z = \alpha W_1 z + (1-\alpha) W_{1,+} z.$$

In the rest of the proof we assume the input vectors are in the block form as well as the weight matrices and denote them in boldface. This does not change the operation of the neural network. Next we prove a central technical lemma which concerns concentration of the matrix $W_{1,+}^{T} W_{1,+} z$. First we define the following matrices. For any nonzero $x, z \in \mathbb{R}^n$ with an angle $\theta_{x,z} = \angle(x, z)$ between them, let

$$Q_{x,z} := \frac{\pi - \theta_{x,z}}{2\pi} \text{Id}_n + \frac{\sin \theta_{x,z}}{2\pi} M_{x \leftrightarrow z}. \quad (B.1)$$

Similarly for two block vectors $x = [x_i]_{i=1}^{n}$ and $z = [z_i]_{i=1}^{n}$, define the block matrix

$$Q_{x,z} := [Q_{x_i,z_i}]_{i=1}^{n}. \quad (B.2)$$

The following result appears in [188].

**Lemma 34 (Lemma 5 [188]).** Fix $\epsilon \in (0,1)$. Let $A \in \mathbb{R}^{n \times k}$ have i.i.d. $\mathcal{N}(0,1/n)$ entries. If $n > c k \log k$, then with probability at least $1 - 8ne^{-\gamma k}$,

$$\forall x, z \in \mathbb{R}^k, \|A_{+}^{T} A_{+} - Q_{x,z}\| \leq \epsilon. \quad (B.3)$$

When $x = z$, it holds

$$\forall x \neq 0, \|A_{+}^{T} A_{+} - \text{Id}_n/2\| \leq \epsilon. \quad (B.4)$$

Here $\epsilon, \gamma$ depends only on $\epsilon$.

Here the matrix $Q_{x,z}$ happens to be the expectation of the matrix $A_{+}^{T} A_{+}$. This can be shown by an elementary calculation. We can now state the central technical lemma.

**Lemma 35.** Fix $\epsilon \in (0,1)$. Let $W = [W_{i}]_{i=1}^{D_0}$ where $W \in \mathbb{R}^{C_1 \ell \times C_0}$ have i.i.d. $\mathcal{N}(0,1/C_1 \ell)$. If $C_1 \ell \gtrsim C_0 \log C_0$, then with probability at least $1 - 8D_0 \ell C_1 e^{-\gamma C_0}$,

$$\forall x, z \in \mathbb{R}^{C_0 D_0}, \|W_{+}^{T} W_{+} - Q_{x,z}\| \leq \epsilon. \quad (B.5)$$
When \( x = y \), it holds
\[
\forall x \neq 0, \| W_{+,x}^T W_{+,x} - I/2 \| \leq \epsilon. \tag{B.6}
\]

Here \( \epsilon, \gamma \) depends only on \( \epsilon \).

Lemma 35 is crucial to the rest of the proof. We note that the conditions of Theorem 27 are almost identical to the ones of Lemma 35. In other words, the concentration of weight matrices as given in (4.4) and (4.5) are enough to imply the existence of a strict descent direction for the cost function \( J(x) \) outside of two small neighborhoods. We now prove the lemma.

**Proof.** Observe that for block vectors \( x = [x_i]_{i=1}^{D_0} \) and \( z = [z_i]_{i=1}^{D_0} \) we can write
\[
W_{1,+,x}^T W_{1,+,x} = [W_{+,x_i}^T W_{+,z_i}]_{i=1}^{D_0}. \tag{B.7}
\]

We know from spectral norm of block matrices that
\[
\| W_{1,+,x}^T W_{1,+,x} - Q_{x,x} \| = \max_{i=1,\ldots,D_0} \| W_{+,x_i}^T W_{+,z_i} - Q_{x_i,z_i} \|. \tag{B.8}
\]

Lemma 34 implies that if \( C_1 \ell \gtrsim C_0 \log C_0 \) then with probability at least \( 1 - 8C_1 \ell e^{-\gamma C_0} \), it holds that \( \| W_{1,+,x}^T W_{+,z} - Q_{x,z} \| \leq \epsilon \). A union bound argument yields
\[
P(\| W_{1,+,x}^T W_{1,+,x} - Q_{x,x} \| \leq \epsilon) \leq \sum_{i=1}^{D_0} P(\| W_{+,x_i}^T W_{+,z_i} - Q_{x_i,z_i} \| \leq \epsilon) \leq 8D_0 C_1 \ell e^{-\gamma C_0}.
\]

Next, we present another useful result that controls how \( W_{1,+,x} \) distorts the angle between two vectors \( x, z \). Let:
\[
g(\theta) := \cos^{-1}\left(\frac{(\pi - \theta) \cos \theta + \sin \theta}{\pi}\right).
\]

First we borrow another result from [188].

**Lemma 36** (Lemma 23 [188]). Fix \( \epsilon \in (0, 0.1) \). Let the conditions of Lemma 34 hold and \( A \) satisfy (B.3). For \( x, z \) denote \( \theta_0 = \angle(x, z) \) and \( \theta_1 := \angle(A_{+,x}x, A_{+,z}z) \). Then
\[
| \theta_1 - g(\theta_0) | \leq 4\sqrt{\epsilon}.
\]

This lemma shows that a Gaussian matrix combined with ReLU operation preserves the angle between vectors up to function \( g(\cdot) \). Our next lemma uses this result.

**Lemma 37.** Fix \( \epsilon < 1/(16\pi)^2 \). Assume that weight matrices \( W_1 \) and \( W_2 \) satisfy (B.5) with constant \( \epsilon \). Then it holds for all \( x, z \neq 0 \) that
\[
\langle W_{2,+,x}^T W_{1,+,x} x, W_{2,+,z} W_{1,+,z} z \rangle > 0.
\]
Proof. We operate under the assumptions that the weight matrices $W_j$ for layer $i = 1, 2$ satisfy Equation B.4 and Equation B.6. In particular these equations imply that for layer $i = 1, 2$ and all input vector $z, z \neq 0$,

\[
\frac{1}{2} - \epsilon \leq \|W_{i,z}\|^2 \leq \frac{1}{2} + \epsilon \tag{B.9}
\]

\[
\frac{1}{2} - \epsilon \leq \|W_{i,z}\|^2 \leq \frac{1}{2} + \epsilon. \tag{B.10}
\]

Since $z^{(1)} = W_{1,z}z^{(0)}$, it follows that for all blocks $j = 1, \ldots, D_0$ that

\[
\sqrt{\frac{1}{2} - \epsilon} \leq \|z_j^{(0)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z_j^{(1)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z_j^{(0)}\|_2 \quad \tag{B.11}
\]

The same statements hold true for $x$ and $x_j^{(i)}$ as well. Consequently, by dividing the inequalities we have for $x$ and $z$ that

\[
\sqrt{\frac{1}{2} - \epsilon} \leq \|x_j^{(0)}\|_2 \leq \|x_j^{(1)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z_j^{(0)}\|_2 \leq \|z_j^{(1)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z_j^{(0)}\|_2 \quad \tag{B.12}
\]

\[
\sqrt{\frac{1}{2} - \epsilon} \leq \|x^{(0)}\|_2 \leq \|x^{(1)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z^{(0)}\|_2 \leq \|z^{(1)}\|_2 \leq \sqrt{\frac{1}{2} + \epsilon} \leq \|z^{(0)}\|_2 \quad \tag{B.13}
\]

We assume without loss of generality that the input vectors $x^{(0)} = [x_j^{(0)}]_{j=1}^{D_0}$ and $z^{(0)} =$
\( [z_j^{(0)}]_{j=1}^{D_0} \) are block normalized, i.e., \( \|x_j^{(0)}\|_2 = \|z_j^{(0)}\|_2 = 1 \). We have
\[
\langle x^{(1)}, z^{(1)} \rangle = \sum_j \langle x_j^{(1)}, z_j^{(1)} \rangle = \|x^{(1)}\|_2 \|z^{(1)}\|_2 \cos \theta^{(1)} = \sum_j \|x_j^{(1)}\|_2 \|z_j^{(1)}\|_2 \cos \theta_j^{(1)}
\]
\[
\geq \min_j \cos \theta_j^{(1)} \sum_j \|x_j^{(1)}\|_2 \|z_j^{(1)}\|_2.
\]
(B.14)

Using the fact that input vectors are block normalized, it follows from the first inequality in (B.12) that
\[
\sqrt{\frac{1 - 2\epsilon}{1 + 2\epsilon}} \|z_j^{(1)}\|_2^2 \leq \|x_j^{(1)}\|_2 \|z_j^{(1)}\|_2
\]
and from the second inequality in (B.13) that
\[
\|x^{(1)}\|_2 \|z^{(1)}\|_2 \leq \sqrt{\frac{1 + 2\epsilon}{1 - 2\epsilon}} \|z^{(1)}\|_2^2
\]
(B.16)

Combining (B.14), (B.15) and (B.16) yields that
\[
\sqrt{\frac{1 + 2\epsilon}{1 - 2\epsilon}} \|z^{(1)}\|_2 \cos \theta^{(1)} \geq \sqrt{\frac{1 - 2\epsilon}{1 + 2\epsilon}} \min_j \cos \theta_j^{(1)} \sum_j \|z_j^{(1)}\|_2^2
\]
(B.17)

which in turn implies that
\[
(1 + 8\epsilon) \cos \theta^{(1)} \geq \min_j \cos \theta_j^{(1)} = \cos \theta^{(1)},
\]
(B.18)

for some \( \hat{i} \). Here we used the fact that \( \frac{1 + \epsilon}{1 - \epsilon} \leq 1 + 4\epsilon \) for \( 0 \leq \epsilon \leq \frac{1}{2} \). Lemma 36 implies that \( |\theta_i^{(1)} - g(\theta_i^{(0)})| \leq 4\sqrt{\epsilon} \). Since \( g(\theta_i^{(0)}) < \frac{\pi}{4} \) (see Figure B-1), we have \( 0 \leq \theta_i^{(1)} \leq \frac{\pi}{4} + 4\sqrt{\epsilon} \). Then for small enough \( \epsilon \), (B.18) implies that
\[
\cos \theta^{(1)} \geq \frac{\cos \theta_i^{(1)}}{1 + 8\epsilon} \geq \frac{\cos \left( \frac{\pi}{4} + 4\sqrt{\epsilon} \right)}{1 + 8\epsilon} > 0.6.
\]
(B.19)

The last inequality comes from the fact that \( \frac{\cos \left( \frac{\pi}{4} + 4\sqrt{\epsilon} \right)}{1 + 8\epsilon} \) is monotonically decreasing with small \( \epsilon \), so the minimum can be computed and is roughly equals to 0.646, since \( \epsilon < 1/(16\pi)^2 \). The exact number doesn’t affect the final results, because we only want to bound \( \cos \theta^{(1)} \) away from 0.

Now we proceed to the second layer of the network. As explained in the Section B.1, we first reorder vectors \( \tilde{x}^{(1)} = [\tilde{x}_j^{(1)}]_{j=1}^{D_1} \) and \( \tilde{z}^{(1)} = [\tilde{z}_j^{(1)}]_{j=1}^{D_1} \). We expand the dot product of \( x^{(1)} \) and \( z^{(1)} \) similarly as before
\[
\|x^{(1)}\|_2 \|z^{(1)}\|_2 \cos \theta^{(1)} = \sum_j \|\tilde{x}_j^{(1)}\|_2 \|\tilde{z}_j^{(1)}\|_2 \cos \tilde{\theta}_j^{(1)}.
\]
Define the set of indices
\[ I = \{ j : \| \tilde{x}_j^{(1)} \|_2 \neq 0 \text{ and } \| \tilde{z}_j^{(1)} \|_2 \neq 0 \}. \]

Then we can continue
\[
\| x^{(1)} \|_2 \| z^{(1)} \|_2 \cos \theta^{(1)} = \sum_{j \in I} \| \tilde{x}_j^{(1)} \|_2 \| \tilde{z}_j^{(1)} \|_2 \cos \tilde{\theta}_j^{(1)} \\
\leq \max_{j \in I} \cos \tilde{\theta}_j^{(1)} \sum_{j \in I} \| \tilde{x}_j^{(1)} \|_2 \| \tilde{z}_j^{(1)} \|_2 \\
\leq \max_{j \in I} \cos \tilde{\theta}_j^{(1)} \| \tilde{x}^{(1)} \|_2 \| \tilde{z}^{(1)} \|_2
\]

where we used Cauchy-Schwartz inequality in the last line. Since the reordering does not change the norm of the block vectors, we have \( \| x^{(1)} \|_2 \| z^{(1)} \|_2 = \| \tilde{x}^{(1)} \|_2 \| \tilde{z}^{(1)} \|_2 \). Consequently, it follows that \( \cos \theta^{(1)} \leq \max_{j \in I} \cos \tilde{\theta}_j^{(1)} \). In other words, for some \( \tilde{j} \in I \),
\[
0.6 < \cos \theta^{(1)} \leq \cos \tilde{\theta}_j^{(1)}. \quad (B.20)
\]

Recall that after the second transposed convolution layer of the network, we have \( z_j^{(2)} = W_2 \tilde{z}_j^{(1)} \). Since similar relation as in (B.11) holds for second layer as well and we have \( \| \tilde{x}_j^{(1)} \|_2 > 0 \) and \( \| \tilde{z}_j^{(1)} \|_2 > 0 \) as \( \tilde{j} \in I \), it follows that
\[
\| \tilde{x}_j^{(2)} \|_2 > 0 \quad \text{and} \quad \| \tilde{z}_j^{(2)} \|_2 > 0. \quad (B.21)
\]

By invoking Lemma 36 once again, we have \( |\theta_j^{(2)} - g(\tilde{\theta}_j^{(1)})| \leq 4\sqrt{\epsilon} \). Combining this with (B.20) yields
\[
\cos \theta_j^{(2)} \geq 0.6. \quad (B.22)
\]

Finally we arrive at the desired result
\[
\langle W_{2,+}x W_{1,+}x^{(0)}, W_{2,+}x W_{1,+}x z^{(0)} \rangle = \langle x^{(2)}, z^{(2)} \rangle = \sum_j \| x_j^{(2)} \|_2 \| z_j^{(2)} \|_2 \cos \theta_j^{(2)} \\
\geq \| x_j^{(2)} \|_2 \| z_j^{(2)} \|_2 \cos \theta_j^{(2)} > 0
\]

which follows from (B.21) and (B.22).

\[ \square \]

**B.3  Additional Lemmas**

Recall that the block weight matrix is defined as \( W_1 = [W_1]_{j=1}^{D_1} \) where \( W_1 \in \mathbb{R}^{C_1 \times C_0} \). Similarly, \( W_2 = [W_2]_{j=1}^{D_1} \) where \( W_2 \in \mathbb{R}^{C_2 \times C_1} \). In the following lemma, we extend the
Also, from Equation B.10 we have
\[
\mathcal{h} \nonumber
\]
Using these two along with the triangle inequality, it follows that
\[
\|W_{1,+}^T W_{2,+}^T W_{2,+} W_{1,+} - I/4\| \leq 2\epsilon. \tag{B.23}
\]

**Proof.** From Lemma 35, we know that for all \( z \neq 0 \),
\[
\|W_{1,+}^T W_{1,+} - I/2\| \leq \epsilon \\
\|W_{2,+}^T W_{2,+} - I/2\| \leq \epsilon.
\]
Also, from Equation B.10 we have
\[
\|W_{1,+}\|^2 \leq \frac{1}{2} + \epsilon.
\]

Using these two along with the triangle inequality, it follows that
\[
\|W_{1,+}^T W_{2,+}^T W_{2,+} W_{1,+} - I/4\|
\]
\[
= \left\| \left( W_{1,+}^T W_{2,+} W_{2,+} W_{1,+} - \frac{1}{2} W_{1,+} W_{1,+} \right) + \left( \frac{1}{2} W_{1,+} W_{1,+} - I/4 \right) \right\|
\]
\[
\leq \|W_{1,+}^T (W_{2,+} W_{2,+} - I/2) W_{1,+}\| + \frac{1}{2} \|W_{1,+}^T W_{1,+} - I/2\|
\]
\[
\leq \|W_{1,+}\|^2 \|W_{2,+} W_{1,+} - I/2\| + \frac{1}{2} \|W_{1,+} W_{1,+} - I/2\|
\]
\[
\leq \left( \frac{1}{2} + \epsilon \right) \epsilon + \frac{1}{2} \epsilon = \epsilon + \epsilon^2 < 2\epsilon.
\]

Given block vectors \( \mathbf{x} \) and \( \mathbf{z} \), we define the following vector
\[
\tilde{h}_{\mathbf{x},\mathbf{z}} := \left[ \frac{\pi - \tilde{\theta}_j^{(1)}}{2\pi} \right]_{j=1}^{D_1} \odot \left[ \frac{\pi - \tilde{\theta}_j^{(0)}}{2\pi} \right]_{j=1}^{D_0} \odot \mathbf{z} + \left[ \frac{\sin \tilde{\theta}_j^{(0)} ||\tilde{\mathbf{z}}_j^{(0)}||_2}{2\pi ||\tilde{\mathbf{z}}_j^{(0)}||_2} \right]_{j=1}^{D_0} \odot \mathbf{x} + \left[ \frac{\sin \tilde{\theta}_j^{(1)} ||\tilde{\mathbf{z}}_j^{(1)}||_2}{4\pi ||\tilde{\mathbf{z}}_j^{(1)}||_2} \right]_{j=1}^{D_1} \odot \mathbf{x} \tag{B.24}
\]
where \( \tilde{\theta}_j^{(1)}, \tilde{x}_j^{(1)}, \tilde{z}_j^{(1)} \) are defined as in Section B.1 and Table B.1. Next we show that the vector \( W_{1,+}^T W_{2,+}^T W_{2,+} W_{1,+} \mathbf{z} \) concentrates around this random vector \( \tilde{h}_{\mathbf{x},\mathbf{z}} \).

**Lemma 39.** Assume \( W_1 \) and \( W_2 \) satisfy the conditions of Lemma 35. Then for all \( \mathbf{x} \neq 0, \mathbf{z} \neq 0 \) we have
\[
\|W_{1,+}^T W_{2,+}^T W_{2,+} W_{1,+} \mathbf{z} - \tilde{h}_{\mathbf{x},\mathbf{z}}\|_2 \lesssim \epsilon \max\{ ||\mathbf{x}||_2, ||\mathbf{z}||_2 \}.
\]
Proof. We expand
\[
W_{1,+}^T W_{2,+}^T W_{1,+}z = W_{1,+}^T W_{2,+}z - Q_{x(1),x(1)} W_{1,+}z
\]
\[
+ \left[ \frac{\pi - \theta^{(1)}_j}{2\pi} \right]_{j=1}^{D_1} \circ W_{1,+}^T W_{1,+}z + \left[ \frac{\sin \theta^{(1)}_j \|z^{(1)}_j\|_2}{2\pi \|z^{(1)}_j\|_2} \right]_{j=1}^{D_1} \circ W_{1,+}^T W_{1,+}x.
\]

We have
\[
|T_1| \leq \|W_{1,+}x\| \|W_{1,+}z\| \|W_{2,+}z - Q_{x(1),x(1)}\| \|z\|_2 \leq \epsilon \|z\|_2 \tag{B.25}
\]
where we used (B.10) and Lemma 35. Expanding \(T_2\) we get
\[
T_2 - \left[ \frac{\pi - \theta^{(1)}_j}{2\pi} \right]_{j=1}^{D_1} \circ Q_{x,z} = \left[ \frac{\pi - \theta^{(1)}_j}{2\pi} \right]_{j=1}^{D_1} \circ \left[ W_{1,+}^T W_{1,+}z - Q_{x,z} \right]
\]
which can be bounded as follows
\[
|T_2 - \left[ \frac{\pi - \theta^{(1)}_j}{2\pi} \right]_{j=1}^{D_1} \circ Q_{x,z} | \leq \|W_{1,+}^T W_{1,+}z\| \|z\|_2 \leq \epsilon \|z\|_2 \tag{B.26}
\]
where we used (B.5). Using the definition (B.1) of \(Q_{x,z}\) we observe that
\[
Q_{x,z} = \left[ \frac{\pi - \theta^{(0)}_j}{2\pi} \right]_{j=1}^{D_0} \circ z + \left[ \frac{\sin \theta^{(0)}_j \|z^{(0)}_j\|_2}{2\pi \|z^{(0)}_j\|_2} \right]_{j=1}^{D_0} \circ x. \tag{B.27}
\]

Expanding \(T_3\) we get
\[
|T_3 - \left[ \frac{\sin \theta^{(1)}_j \|z^{(1)}_j\|_2}{4\pi \|z^{(1)}_j\|_2} \right]_{j=1}^{D_1} \circ x | = \left[ \frac{\sin \theta^{(1)}_j \|z^{(1)}_j\|_2}{2\pi \|z^{(1)}_j\|_2} \right]_{j=1}^{D_1} \circ \left[ W_{1,+}^T W_{1,+} - I/2 \right] \| x \|_2 \leq \epsilon \| x \|_2 \tag{B.28}
\]
where we used (B.6) in the last inequality. The result follows by combining (B.25), (B.26), (B.27) and (B.28).

Before we proceed to finishing the proof of Theorem 27, we introduce a couple more definitions and lemmas:

\[
h_{z,x^o} = \frac{z}{4} - \tilde{h}_{z,x^o} \tag{B.29}
\]

\[
S_{\epsilon,x^o} = \{ z : \|h_{z,x^o}\|_2 \leq \epsilon \max(\|z\|_2, \|z^o\|_2) \} \tag{B.30}
\]

Note that \(\tilde{h}_{z,x^o}\) has already been defined in Equation B.24, and \(h_{z,x^o}\) is simply the perturbation of \(\tilde{h}_{z,x^o}\) around its mean. \(S_{\epsilon,x^o}\) is the region where such perturbation is small.
Lemma 40. Assume the conditions of Lemma 38 and Lemma 39 hold. Then, with high probability for all \( z \neq 0 \),

\[
\left\| v_{z, x^\circ} - h_{z, x^\circ} \right\|_2 < 2\epsilon \max(\|z\|_2, \|z^\circ\|_2)
\] (B.31)

Before the proof for Lemma 40, we introduce a descent direction \( v_{z, x^\circ} \) and some notation:

\[
v_{z, x^\circ} = (W_{2,+} W_{1,+})^T (W_{2,+} W_{1,+}) z - (W_{2,+} W_{1,+})^T (W_{2,+} x^\circ W_{1,+}) z^\circ
\]

\[= T_1 z - T_2 z^\circ\] (B.32)

Proof for Lemma 40. Under the conditions of Lemma 38, with high probability we have that

\[
\|T_1 - I/4\| \leq 2\epsilon.
\] (B.33)

In addition, under the conditions of Lemma 39, with high probability and a change of notation we have that

\[
\left\| T_2 z^\circ - \tilde{h}_{z, x^\circ} \right\|_2 \leq \epsilon \|z^\circ\|_2.
\] (B.34)

Combining the definition of \( h_{z, x^\circ} \) and Equation B.32, it follows that

\[
\left\| v_{z, x^\circ} - h_{z, x^\circ} \right\|_2 \leq \left\| (T_1 z - T_2 z^\circ) - (z^\circ - \tilde{h}_{z, x^\circ}) \right\|_2
\]

\[
\leq \left\| T_1 z - \frac{1}{4} z \right\|_2 + \left\| T_2 z^\circ - \tilde{h}_{z, x^\circ} \right\|_2
\]

\[
\leq \|T_1 - I/4\|_2 \|z\|_2 + \left\| T_2 z^\circ - \tilde{h}_{z, x^\circ} \right\|_2
\]

\[
\leq 2\epsilon \|z\|_2 + \epsilon \|z^\circ\|_2
\]

\[
< 2\epsilon \max(\|z\|_2, \|z^\circ\|_2)
\]

The first inequality applies triangle inequality, the second uses the simple fact that \( \|Ax\| \leq \|A\| \|x\|_2 \), and the last inequality is a result of both Line B.33 and Line B.34. \( \square \)

Now we introduce the last lemma needed for the main proof. This lemma establishes that the set \( S_{\epsilon, z} \) is a subset of the union of two small neighbors around \( z^\circ \) and \( -\rho z^\circ \) with radius no more than \( \epsilon \).

**Lemma 41.** Suppose \( 4\epsilon \leq 1 \). Then

\[
S_{4\epsilon, z^\circ} \subset B(z^\circ, c\epsilon \|z^\circ\|_2) \cup B(-\rho z^\circ, d\epsilon \|z^\circ\|_2)
\]

where \( \rho, c, d \) are universal constants.

**Proof.** The proof mainly follows Lemma 8 in [188]. Using the definitions (B.24) and (B.29)
we can rewrite $h_{z,z^o}$ as follows:

$$h_{z,z^o} = -\beta z^o + (1/4 - \alpha)z$$

where the vectors $\alpha$ and $\beta$ are

$$\alpha = \left[ \frac{\pi - \theta_j^{(1)}}{2\pi} \right] \sum_{j=1}^{D_1} \left[ \frac{\sin \theta_j^{(0)} \| z_j^{(0)} \|_2}{2\pi \| z_j^{(0)} \|_2} \right] D_0, \left[ \frac{\sin \theta_j^{(1)} \| z_j^{(1)} \|_2}{4\pi \| z_j^{(1)} \|_2} \right] D_1 \right] \sum_{j=1}^{D_1},$$

$$\beta = \left[ \frac{\pi - \theta_j^{(1)}}{2\pi} \right] \sum_{j=1}^{D_1} \left[ \frac{\pi - \theta_j^{(0)}}{2\pi} \right] D_0 \right] \sum_{j=1}^{D_1}.$$

Without loss of generality, let $\| z_\downarrow^{(0)} \|_2 = 1$, $z \in S_{\epsilon,x^o}$ and $M = \max(\max_j \| z_j^{(0)} \|_2, 1)$. Then definition of $S_{\epsilon,x^o}$ in (B.30) implies that $\| h_{z,z^o} \|_2$ is small. This, in turn, implies that

$$| - \beta_j + \cos \theta_j^{(0)} (1/4 - \alpha_j) | \leq \epsilon M, \quad (B.35)$$

$$| \sin \theta_j^{(0)} (1/4 - \alpha_j) | \leq \epsilon M \text{ for all } j. \quad (B.36)$$

Recall that $\theta_j^{(0)}$ is the angle between $z_j^{(0)}$ and $z_j^{(0)}$, which are the $j$-th block of the vectors $z$ and $z^o$ respectively. For the rest of the proof, we argue as follows. Using (B.35) and (B.36), we will show that for all $j$ either

$$\theta_j^{(0)} \leq 2\epsilon \text{ and } \| z_j^{(0)} \|_2 \simeq \| z_j^{(0)} \|_2 \quad (B.37)$$

or

$$| \theta_j^{(0)} - \pi | \leq 2\epsilon \text{ and } \| z_j^{(0)} \|_2 \simeq \rho \| z_j^{(0)} \|_2 \quad (B.38)$$

holds for some constant $\rho$. This, in turn, implies that for all $j$, $z_j^{(0)}$ is either very close to $z_j^{(0)}$ or to the polar opposite of it up to a constant $\rho$. Indeed, observe the fact that

$$\| z_j^{(0)} - z_j^{(0)} \|_2 \leq \| z_j^{(0)} \|_2 - \| z_j^{(0)} \|_2 + \left( \| z_j^{(0)} \|_2 + \| z_j^{(0)} \|_2 - \| z_j^{(0)} \|_2 \right) \theta_j^{(0)}.$$

This simply says that if a vector is known to have magnitude within $\Delta r$ of some other vector with magnitude $r$ and two vectors have an angle less than $\Delta \theta$ between them, then the Euclidean distance between two vectors is no more than $\Delta r + (r + \Delta r)\Delta \theta$. Then using the assumption $\| z_j^{(0)} \|_2 = 1$ and (B.37) we have

$$\| z_j^{(0)} - z_j^{(0)} \|_2 \leq \epsilon.$$

Given this, we can conclude that

$$\| z - z^o \|_2^2 = \sum_{j=1}^{D_0} \| z_j^{(0)} - z_j^{(0)} \|_2 \leq \epsilon^2 D_0$$

139
which implies \( z \in B(z^\circ, c \| z^\circ \|_2) \) for some constant \( c \). A similar argument holds for \( z_j^{(0)} \) and \( \rho z_j^{\circ,(0)} \) using (B.38). This implies the claim of the lemma.

We now proceed to prove our claim that either (B.37) or (B.38) holds. We make the following observations:

\[
|\beta_j| \leq 1, \quad (B.39)
\]
\[
|\alpha_j| \leq \frac{1 + \sin \theta_j^{(0)}}{2\pi \| z_j^{(0)} \|_2} \quad (B.40)
\]

for all \( j \). Following similar arguments from [188, Proof of Lemma 8] and using (B.39) and (B.40), we can deduce that \( M \leq 2 \). Hand et.al. in [188] also establish using \( M \leq 2 \), (B.39) and (B.40) that for all \( j \), \( \theta_j^{(0)} \) is either small, i.e., \( \theta_j^{(0)} \approx 0 \), or large, i.e., \( \theta_j^{(0)} \approx \pi \). We refer to [188, Proof of Lemma 8] for details. Now we consider those two cases.

**Small angle case:** Recall that we eventually aim to show either that (B.37) or (B.38) holds. Therefore under the assumption that \( \theta_j^{(0)} \) is small, it is enough to show that \( \| z_j^{(0)} \|_2 \approx 1 \). Assume that \( \theta_j^{(0)} = O(\epsilon) \). Then \( \sin \theta_j^{(0)} = O(\epsilon) \) and \( \cos \theta_j^{(0)} = 1 + O(\epsilon) \).

Combining these with \( M \leq 2 \), (B.35), (B.39) and (B.40) yields \( \| z_j^{(0)} \|_2 \approx 1 + O(\epsilon) \) which is the desired result.

**Large angle case:** Assume that \( \theta_j^{(0)} = \pi + O(\epsilon) \). Similarly as before, it suffices to show that \( \| z_j^{(0)} \|_2 \approx \rho \) for some constant \( \rho \). We have \( \sin \theta_j^{(0)} = O(\epsilon) \) and \( \cos \theta_j^{(0)} = -1 + O(\epsilon) \).

Combining these with \( M \leq 2 \), (B.35), (B.39) and (B.40) yields \( \| z_j^{(0)} \|_2 \approx \frac{2}{6\pi} = \rho \). This completes the proof.

In the remaining we make the following abbreviations: \( v_z = v_{z,z^\circ} \), \( h_z = h_{z,z^\circ} \), and \( S_z = S_{z,z^\circ} \).

### B.4 Finishing the proof for Theorem 27

We first prove Equation 4.5 of Theorem 27, i.e., the existence of local maximum at the origin. Recall that the descent direction \( v_{z,z^\circ} \) is already given in Equation B.32, which we repeat as follows:

\[
v_{z,z^\circ} = (W_{2,+z} W_{1,+z})^T (W_{2,+z} W_{1,+z}) z - (W_{2,+z} W_{1,+z})^T (W_{2,+z} W_{1,+z}) z^\circ.
\]

This expression is the gradient of \( J \) where \( J \) is differentiable. By using the definition of one-sided directional derivative \( D_z \) compute that

\[
D_z J(0) = -\langle W_{2,+z} W_{1,+z} z, W_{2,+z} W_{1,+z} z^\circ \rangle \leq -\frac{1}{16\pi} \| z \|_2 \| z^\circ \|_2 = 0.
\]

\[
140
\]
where the inequality follows from Lemma 37 and the second inequality follows since $z, z^\circ \neq 0$.

Proof of Equation (4.4) is more involved. It mainly follows the arguments in [188](Section 6.1) adapted to our block structure of the weight matrices. One needs to eventually compute the directional derivative $D_{v_z}z$ and show that it is negative when $z$ is outside of two neighborhoods given in (4.4). Lemma 41 establishes that $S_{1c}$ is a subset of two small neighborhoods around $z^\circ$ and $-\rho z^\circ$. Therefore, it is sufficient to show that for all $z$ outside $S_{1c}$, the derivative along the descent direction $v_z$ is always negative. By definition, the (unnormalized) one-sided directional derivative of $J(z)$ in the direction of $v_z$ at $z$ is

$$D_{v_z}J(z) = \lim_{t \to 0^+} \frac{J(z + tv_z) - J(z)}{t}.$$  

Since the function $\sigma(W_2\sigma(W_1z))$ is both continuous and piecewise linear, it follows that for any $z \neq 0$ and $v_z \neq 0$ there exists a sequence $\{z_n\} \to z$ such that $J$ is differentiable at each $z_n$, and $D_{v_z}J(z) = \lim_{n \to \infty} \nabla J(z_n) \cdot v_z$. Since $\nabla J(z_n) = v_{z_n}$, it follows that

$$D_{v_z}J(z) = \lim_{n \to \infty} v_{z_n} \cdot v_z.$$  

Now we bound the right-hand side from below

$$v_{z_n} \cdot v_z = h_{z_n} \cdot h_z + (v_{z_n} - h_{z_n}) \cdot h_z + h_{z_n} \cdot (v_z - h_z) + (v_{z_n} - h_{z_n}) \cdot (v_z - h_z) \geq h_{z_n} \cdot h_z - \|v_{z_n} - h_{z_n}\|_2 \|h_z\|_2 - \|v_z - h_z\|_2 \|v_{z_n} - h_{z_n}\|_2 \|v_z - h_z\|_2$$

$$\geq h_{z_n} \cdot h_z - \epsilon \max(\|z_n\|_2, \|z^\circ\|_2) \|h_z\|_2$$

$$- \epsilon \max(\|z\|_2, \|z^\circ\|_2) \|h_{z_n}\|_2 - \epsilon^2 \max(\|z_n\|_2, \|z^\circ\|_2) \max(\|z\|_2, \|z^\circ\|_2),$$

where the first inequality follows from the triangle inequality and the second is a result of Lemma 40.

As $h_{z_n}$ is continuous in $z$ for all nonzero $z$, it follows that for any $z \notin S_{1c}$,

$$\lim_{n \to \infty} v_{z_n} \cdot v_z \geq \|h_z\|_2^2 - 2\epsilon \|h_z\|_2 \max(\|z\|_2, \|z^\circ\|_2) - \epsilon^2 \max(\|z\|_2, \|z^\circ\|_2)^2$$

$$\geq \frac{\|h_z\|_2^2}{2} \left[ \|h_z\|_2^2 - 4\epsilon \max(\|z\|_2, \|z^\circ\|_2) \right] + \frac{1}{2} \left[ \|h_z\|_2^2 - 2\epsilon^2 \max(\|z\|_2, \|z^\circ\|_2)^2 \right]$$

$$> 0$$

The second inequality uses the definition of $S_{1c}$. Consequently, $D_{v_z}J(z) = \lim_{n \to \infty} v_{z_n} \cdot v_z > 0$, and thus $D_{-v_z}J(z) < 0$ for all $z \notin S_{1c}$. Proof is finished by applying Lemma 41.

$\square$
Bibliography


