# Learning to Enhance RGB and Depth Images with Guidance

**Kaiyue Lu**

A thesis submitted for the degree of
Doctor of Philosophy
The Australian National University

January 2022

Draft Copy – 28 January 2022

Except where otherwise indicated, this thesis is my own original work.

Kaiyue Lu
28 January 2022

to my beloved family.

# Acknowledgments

It is approaching the end of my PhD career. I would like to take this opportunity to express my greatest appreciation to those people who support and help me during this long journey.

First of all, I would like to thank my primary/chair supervisor, Prof. Nick Barnes. Nick is my first foreign supervisor in my life. He is a very professional and experienced scholar, and he can always share his excellent ideas with me and give constructive suggestions to my research. He is also very patient in revising my papers, *i.e.*, he highlights all the revisions so that I can learn how to improve the paper writing by comparing against my previous version. Nick is very kind and he can always respect my thinking. He is humorous and easy-going, and I particularly like his laugh, which has the magic of effectively reducing my stress. Indeed, I have learned a lot from his sophisticated and critical thinking on research, and his optimism and kindness. He is my best teacher and best collaborator in my PhD study. I feel very happy when he was promoted as the professor several months ago, and it is a great honor for me to work with him.

I would like to thank my associate supervisor Dr. Liang Zheng. He is the most hard-working teacher I have ever met. His paper writing skill is very excellent, and he can always highlight the most important parts in the paper and summarize them as concrete contributions. He consistently seeks for truth, and always tells me to focus on the fundamental problem itself rather than using fancy descriptions to package methods. I think this is the true essence of research, and I will stick to it forever.

I would like to thank my another associate supervisor Dr. Saeed Anwar. He is my Data61 supervisor, and supports me a lot in using the computational resources there. He is also very professional in revising papers, which can always enlighten me with deeper thinking to the problem. Saeed kindly shares his experience in thesis writing with me, which is quite helpful and valuable.

I would like to give special thanks to my previous supervisor Dr. Shaodi You. Shaodi is the one who recommended me to Data61 via the summer scholar program in late 2016. He also introduced the image smoothing topic to me. Without him, I could not enter Data61 and do my PhD degree. However, something unexpected and unhappy happened in late 2018, and he left my supervisory panel at that time. I fully understood his situation then, but I just wanted to concentrate all my attention on the research. I will always remember him and be grateful to his help in my previous research on image smoothing. Although we have no collaboration now, I wish him good luck on whatever he does from here on out.

I sincerely appreciate the financial support from the Australian National University (ANU) and Data61. ANU exempted my tuition fees and Data61 awarded me

sufficient scholarships. They significantly reduce my daily economic pressure on basic necessities and let me concentrate on the research work. I also acknowledge Data61 that provides me comfortable office and convenient facilities such as computing machines (with GPUs), printers, and meeting rooms.

I would like to thank my initial supervisor Dr. Siyu Xia when I did the bachelor degree in Southeast University. My research dream was started by him, and he was the first supervisor who instructed me how to do research, how to write papers, and how to clearly present works to the audience. All my research achievements can be attributed to him to some extent.

I am also grateful to a number of friends who bring me lots of happiness and support. They are Weixuan Sun, Zipeng Hu, Changkun Ye, Peipei song, Ziang Cheng, Yiran Zhong, Jing Zhang, Hao Zhu, Haoyang Zhang, Chengyue Zhou, Qixin Xu, Liangjun Zhang, Shiyong Tian, and Peng Tao.

I also would like to express my gratefulness and love to my family, including my wife, my son, my parents, my parents-in-law, and my sister. They selflessly provide mental and daily support to me, allowing me to focus on my research and thesis writing. My wife, Lina Cheng, has been looking after our son by herself for the purpose of not distracting my attention. She always tells me to be persistent in research and she will give me consistent understanding and support until I finish it. It is such a great fortune that I met and married her. Lina, I am willing to hold hands with you all my life and age together. I love you!

Last but not least, I want to thank my grandparents and express my deepest missing to them. They both passed away in 2019, at which time I was doing the PhD degree in Australia. Until now, I have been feeling very sad and regretful that I did not spare much time to accompany them. My grandparents brought me up, and dedicated all their love to me. I once made a silent vow in front of their deadee to complete my PhD and never give up. Now I am finishing the study, but can they see it? Can they hear me? Let this thesis dedicated to them. I know they never left me, and they are just watching and waiting for me in the heaven.

# Abstract

Image enhancement improves the visual quality of the input image to better identify key features and make it more suitable for other vision applications. Structure degradation remains a challenging problem in image enhancement, which refers to blurry edges or discontinuous structures due to unbalanced or inconsistent intensity transitions on structural regions. To overcome this issue, it is popular to make use of a guidance image to provide additional structural cues. In this thesis, we focus on two image enhancement tasks, *i.e.*, RGB image smoothing and depth image completion. Through the two research problems, we aim to have a better understanding of what constitutes suitable guidance and how its proper use can benefit the reduction of structure degradation in image enhancement.

Image smoothing retains salient structures and removes insignificant textures in an image. Structure degradation in image smoothing results from the difficulty in distinguishing structures and textures with low-level cues such as gradients and intensity difference. Specifically, structures may be inevitably blurred if the filter tries to remove some strong textures that have high contrast. Moreover, these strong textures may also be mistakenly retained as structures. We address this issue by applying two forms of guidance for structures and textures respectively. We first design a kernel-based double-guided filter (DGF), where we adopt semantic edge detection as structure guidance, and texture decomposition as texture guidance. The DGF is the first kernel filter that simultaneously leverages structure guidance and texture guidance to be both "structure-aware" and "texture-aware". It can remove strong textures without blurring main structures. Considering that textures present high randomness and variations in spatial distribution and intensities, it is not robust to localize and identify textures with hand-crafted features. Hence, we take advantage of deep learning for richer feature extraction and better generalization. Specifically, we generate synthetic data by blending natural textures with clean structure-only images. With the data, we build a texture prediction network (TPN) that estimates the location and magnitude of textures. We then combine the texture prediction results from TPN with a semantic structure prediction network (SPN) so that the final texture and structure aware filtering network (TSAFN) is able to differentiate between structures and textures more effectively and robustly. Our model achieves superior smoothing results than existing filters.

Depth completion recovers dense depth from sparse measurements, *e.g.*, LiDAR. Existing depth-only methods use sparse depth as the only input and suffer from structure degradation, *i.e.*, failing to recover semantically consistent boundaries or small/thin objects due to (1) the sparse nature of depth points and (2) the lack of images to provide structural cues. In the thesis, we attempt to deal with the structure degradation issue by using RGB image guidance in both supervised and unsuper-

vised depth-only settings. For the supervised model (IR), the unique design is that it simultaneously outputs a reconstructed image and a dense depth map. Specifically, we treat image reconstruction from sparse depth as an auxiliary task during training that is supervised by the unlabelled image. For the unsupervised model, we regard dense depth as a reconstructed result of the sparse input, and formulate our model as an auto-encoder (UDAE). To reduce structure degradation, we employ the image to guide latent features by penalizing their difference in the training process. The image guidance loss in both models enables them to acquire more dense and structural cues that are beneficial for producing more accurate and consistent depth values. For inference, the two models only take sparse depth as input and no image is required. On the KITTI Depth Completion Benchmark, we validate the effectiveness of the proposed image guidance through extensive experiments and achieve competitive performance over state-of-the-art supervised and unsupervised methods. Our approach is also applicable to indoor scenes.

# Contents

# List of Figures

# List of Tables

# List of Abbreviation Words

**Image Smoothing**

- **DGF:** double-guided filter
- **TSAFN:** deep texture and structure aware network
- **SPN:** structure prediction network
- **TPN:** texture prediction network
- **BLF:** bilateral filter
- **GF:** guided filter
- **TV:** total variation
- **WLS:** weighted least squares
- **RTV:** relative total variation
- **RGF:** rolling guidance filter
- **SGF:** segment graph filter
- **SDF:** static and dynamic filter
- **SGTD:** structure gradient and texture decorrelating
- **DEAF:** deep edge-aware filter
- **DJF:** deep joint filter
- **DRF:** deep recursive filter
- **DFF:** deep fast filter
- **CEILNet:** cascaded edge and image learning network
- **PSNR:** peak signal to noise ratio
- **SSIM:** structure similarity

**Depth Completion**

- **IR:** depth completion via image reconstruction

- **UDAE:** unsupervised depth completion auto-encoder
- **MTL:** multi-task learning
- **AL:** auxiliary learning
- **SparseConvs:** sparsity invariant convolutional neural networks
- **S2D:** sparse to dense
- **ADNN:** alternating direction neural network
- **CSPN:** convolutional spatial propagation network
- **DDP:** dense depth posterior
- **RMSE:** rooted mean square error
- **MAE:** mean absolute error
- **iRMSE:** inverse rooted mean square error
- **iMAE:** inverse mean absolute error
- **REL:** mean absolute relative error

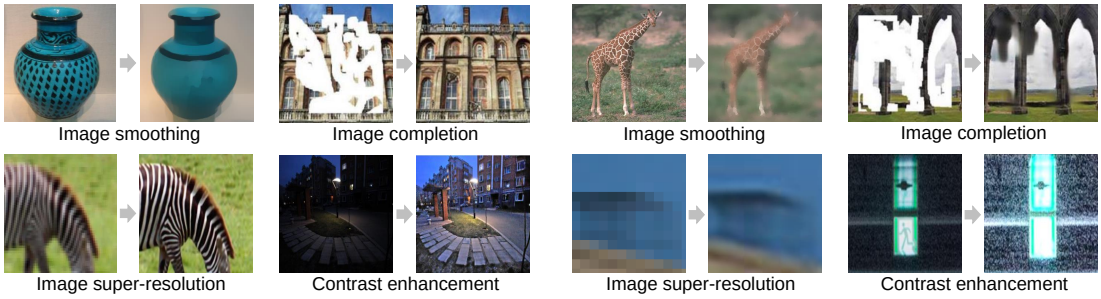
Chapter 1

# Introduction

Image enhancement aims to improve the visual quality of the input image in order to better identify key features and make it more suitable for specific vision tasks [Wang et al., 1983; Shukla et al., 2017], such as edge detection [Bao et al., 2005; Galun et al., 2007], object detection [Zhang et al., 2003; Zhou and Gu, 2018], semantic segmentation [Cho et al., 2020; Wang et al., 2020], and so on. It is an active topic in Digital Image Processing and normally used as a preprocessing technique. In addition to RGB images from cameras, image enhancement has broad applications to images captured from other sensors or devices, *e.g.*, LiDAR depth images [Uhrig et al., 2017], X-ray images [Ahmad et al., 2012], and MRI images in medicine [George and Karnan, 2012].

Image enhancement involves low-level and primitive operations [Gonzalez and Woods, 1977], typically including image smoothing, image completion, image super-resolution, and contrast enhancement. One common goal of these operations is to enhance object structures/edges, which are characterized by sharp intensity transitions on boundaries [Gonzalez and Woods, 1977]. They are essential features for segmenting and recognizing objects. Specifically, *image smoothing* removes insignificant details, *e.g.*, textures and noise, without blurring edges [Tomasi and Manduchi, 1998; He et al., 2013]. *Image completion*, also known as *image inpainting*, fills in missing values or damaged regions and recovers continuous structures based on available pixel values [Komodakis and Tziritas, 2007; He and Sun, 2014]. *Image super-resolution* restores the high-resolution image from the low-resolution input, in which case the sharpness of edges is boosted [Yang, 2010; Dong et al., 2016]. *Contrast enhancement* increases the contrast of the image content for clearer visualization [Abdullah-Al-Wadud et al., 2007; Arici et al., 2009]. Fig. 1.1(a) displays some examples of these tasks.

Both aiming for image quality improvement, image enhancement has some overlap with image restoration in concept and applications. Differently, image restoration focuses more on restoring the original image from its degraded version by assuming a priori knowledge and modelling the degradation [Gonzalez and Woods, 1977], *e.g.*, image denoising [Tao et al., 2000; Burger et al., 2012], deblurring [Katkovnik et al., 2005; Danielyan et al., 2012], dehazing [He et al., 2010; Ancuti and Ancuti, 2013]. Normally, there is an objective comparison between the output and the original, *e.g.*, calculating their difference. By contrast, image enhancement does not need to know




(a) Typical image enhancement tasks (b) Structure degradation

Figure 1.1: Image enhancement. (a) Typical enhancement tasks include image smoothing, image completion, image super-resolution, and contrast enhancement. (b) Structure degradation widely exists in image enhancement, generally referring to blurry edges or discontinuous structures. Both image smoothing examples are from our paper [Lu et al., 2018a]. Both image completion examples are from [Li et al., 2020b]. The left image super-resolution example is from [Hussein et al., 2020], and the right one is from [Liu et al., 2020a]. The left contrast enhancement example is from [Liu et al., 2019], and the right one is from [Chien et al., 2019].

how the input image is corrupted, so it is a more subjective process [Gonzalez and Woods, 1977]. The goodness of the enhanced output is largely dependent on whether the output is visually pleasing to humans. Human judges are primarily concerned with whether object structures are sharp in intensity and continuous in appearance.

However, structure degradation remains a challenging problem in image enhancement. It basically refers to blurry edges or discontinuous structures due to unbalanced or inconsistent intensity transitions around structural regions [Gonzalez and Woods, 1977; Huang and Aizawa, 1993]. For example, in image smoothing, edges are prone to be blurred when the filter tries to remove some strong textures that also have high contrast. Some image completion methods fail to recover consistent structures as too much structural information is missing in the input. It is difficult for super-resolution methods to sharpen edges because the general intensity changes in the low-resolution image are relatively small and flat. Noise amplification widely occurs in contrast enhancement, which inevitably degrades edges and makes them look discontinuous. Structure degradation examples are displayed in Fig. 1.1(b).

To reduce structure degradation, it is popular to make use of a guidance image to provide extra structural cues. We follow [Guo et al., 2020] and classify structure guidance into two categories: (1) self-guidance, and (2) reference-guidance. Self-guidance means the guidance contains certain attributes that are derived from the input image itself, *e.g.*, gradients, intensity difference, edge maps. Reference-guidance is another image that is captured in different conditions, *e.g.*, daytime and night [Guo et al., 2020], flash and non-flash [Petschnigg et al., 2004], or from different modalities, *e.g.*, depth [Kopf et al., 2007], NIR [Yan et al., 2013]. The structural features offered by the guidance are complementary to the original input, so they can improve the overall

performance in enhancing images without degrading structures.

For the use of structure guidance, traditional non-learning methods usually calculate the correlations between the guidance and target images within a local kernel or patch [Tomasi and Manduchi, 1998; He and Sun, 2012; Zhang et al., 2012; Tung and Fuh, 2021], or define a data term (involving each pixel and avoiding significant intensity shift from the input) and a smoothness term (constraining nearby pixels and acting as a regularizer based on guidance) for iterative optimization [Farbman et al., 2008; Komodakis, 2006; Shi et al., 2015; Liu et al., 2019]. They do not need ground truth labels but may suffer from poor robustness to various image content and long processing time. In the deep learning era, Convolutional Neural Networks (CNNs) are able to extract richer features from the guidance image and transfer them into the target image more effectively [Fan et al., 2017b; Yang et al., 2020; Ma et al., 2020; Lv et al., 2021]. These methods are data-driven, and have better robustness and generalization ability after being fully trained with ground truth. Learning to enhance images with guidance has become a new trend.

In this thesis, we focus on two image enhancement tasks, *i.e.*, RGB image smoothing (**image smoothing** for short) and depth image completion (**depth completion** for short). For image smoothing, we study self-guidance, *i.e.*, using semantic structure guidance and proposing a novel texture guidance to improve the discrimination of textures and structures. We design both hand-crafted and deep learning filters for this task. For depth completion, we propose two deep learning models and explore reference-guidance, *i.e.*, the RGB image. We employ the image to facilitate the recovery of structure-consistent dense depth and exploit a new approach for image guidance by incorporating it into the training loss in both supervised and unsupervised settings. Through the two research problems, we aim to have a better understanding of what constitutes suitable guidance and how its proper use can benefit the reduction of structure degradation in image enhancement.

## 1.1 Image Smoothing

### 1.1.1 Background

Image smoothing, a fundamental technology in image processing and computer vision, aims to enhance images by retaining salient structures and removing insignificant textures. In the top left image in Fig. 1.1(a), the vase is covered with black dotted textures. The removal of these textures does not affect the main structure and our recognition of the vase, so they are insignificant details that can be removed. Image smoothing has extensive applications such as denoising [Gu et al., 2014], detail enhancement [Fattal et al., 2007], image abstraction [Winnemöller et al., 2006], image dehazing [Li and Zheng, 2017], and segmentation [Wang and He, 2012].

In image smoothing, structures are generally referred to as large intensity difference between pixels, while textures are small intensity oscillations [Tomasi and Manduchi, 1998; He et al., 2013; Xu et al., 2011; Ham et al., 2017]. Essentially, retaining structures is to keep intensity transitions on edges as sharp as possible. By contrast,

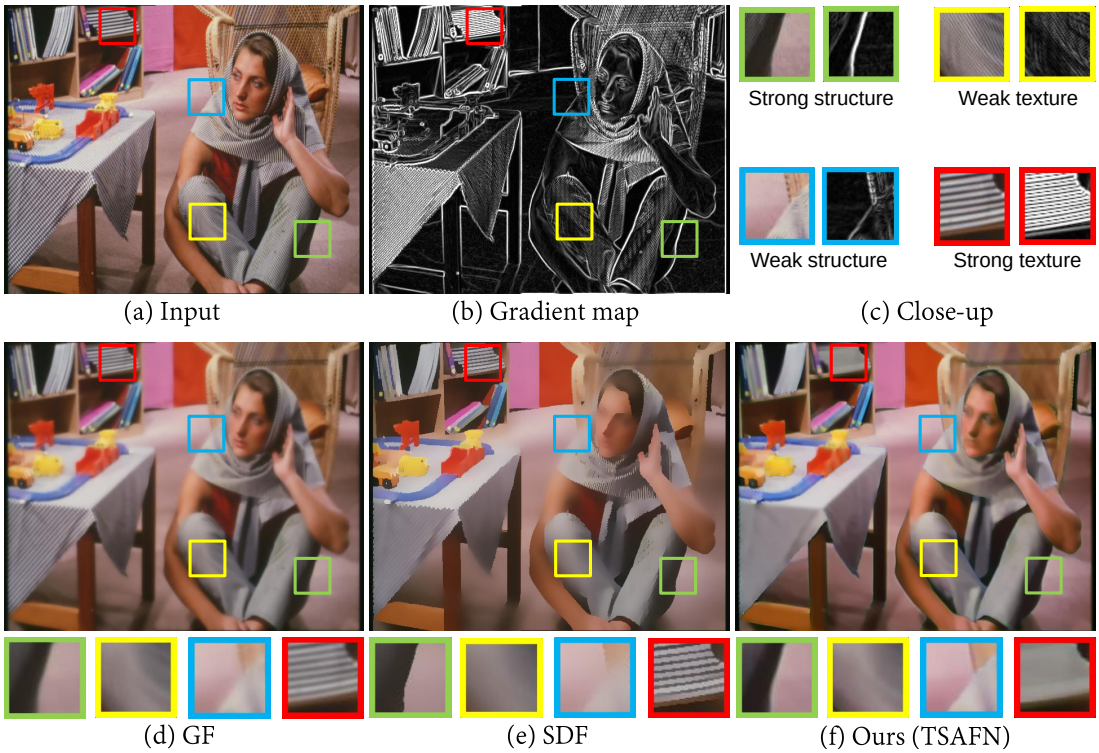



Figure 1.2: Close observation of structures and textures. The general assumption in image smoothing is that structures always have larger gradients (*strong structures*) while the gradients of textures are smaller (*weak textures*). It is easy for existing filters, *e.g.*, GF [He et al., 2013] and SDF [Ham et al., 2017] to preserve strong structures and remove weak textures. However, some strong textures, *e.g.*, stripe textures within books, are either mistakenly retained as edges or suppressed with important structures blurred as a side effect, *e.g.*, weak structures of the arm and chair. Thus, only using gradients cannot effectively differentiate between structures and textures. Our filter, *e.g.*, TSAFN, can remove strong textures without blurring main structures.

removing textures is to flatten the sharpness of intensity changes in texture regions. They are two opposite processes, so image smoothing algorithms have to balance structure preservation and texture removal. There are mainly two types of methods for image smoothing. (1) **Kernel-based methods** that calculate the weighted average of pixel values within a local squared kernel, such as the bilateral filter (BLF) [Tomasi and Manduchi, 1998], the guided filter (GF) [He et al., 2013], and the segment graph filter (SGF) [Zhang et al., 2015]. (2) **Global methods** that decompose the image into a structure layer and a texture layer by optimizing a globally-defined objective function, such as total variation (TV) [Rudin et al., 1992], $\ell_0$ smoothing [Xu et al., 2011], and the static and dynamic guidance filter (SDF) [Ham et al., 2017]. These methods address "structure-awareness" by leveraging a single structure guidance to indicate the location of structures. Structure guidance is mostly derived from hand-crafted features that are largely dependent on low-level cues, *e.g.*, gradients, intensity difference. Here we take gradients for example. For a pixel value $f(x, y)$ in the image, its

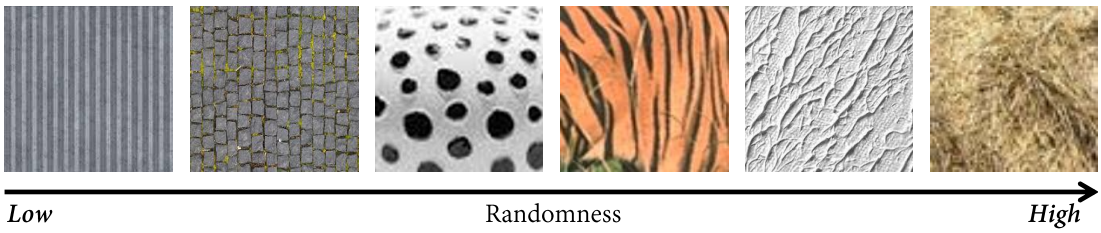



Figure 1.3: Randomness of textures in natural images. Natural textures present various scales with significant spatial distortions and/or color variations. Hand-crafted features cannot robustly and precisely reflect the random nature of textures.

gradient $\mathrm{gd}(f)$ at coordinates $(x, y)$ are defined as the first-order derivatives along $x$ and $y$ directions, *i.e.*,

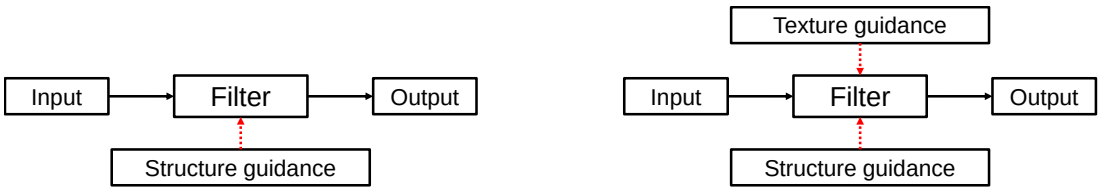
$$\mathrm{gd}(f) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} \approx \begin{bmatrix} f(x+1,y) - f(x,y) \\ f(x,y+1) - f(x,y) \end{bmatrix} = \begin{bmatrix} \mathrm{gd}_x \\ \mathrm{gd}_y \end{bmatrix}. \tag{1.1}$$

The magnitude of the gradient is calculated as

$$m(x,y) = \sqrt{\mathrm{gd}_x^2 + \mathrm{gd}_y^2}. \tag{1.2}$$

For simplicity, in the rest of the thesis, the gradient consistently refers to the magnitude of the gradient unless otherwise specified. The gradient depicts the relative changes in intensity, and existing structure guidance generally assumes that structures always have larger gradients while the gradients of textures are smaller [Guo et al., 2020; Kim et al., 2019b]. However, this assumption is not robust enough to precisely differentiate between structures and textures [Subr et al., 2009; Karacan et al., 2013; Kim et al., 2019b; Fang et al., 2019a; Liu et al., 2020e]. In many cases, structures with relatively small gradients (named *weak structures*) may also have important semantic meaning, and they are prone to be smoothed out as textures. Also, some textures are likely to have large gradients (named *strong textures*), in which case they are either mistakenly retained as edges or suppressed with important structures blurred as a side effect (see Fig. 1.2 for close observation of structures and textures).

In fact, textures are inherently difficult to identify and extract, especially in natural images. This is because textures are essentially repeated patterns regularly or randomly distributed within object structures. They may present various scales with significant spatial distortions and/or color variations (see Fig. 1.3). Hand-crafted features, *e.g.*, intensity difference (or gradients) [Tomasi and Manduchi, 1998], region covariances [Karacan et al., 2013], co-occurrence [Jevnisek and Avidan, 2017], local extrema [Subr et al., 2009], cannot robustly and precisely reflect the random nature of textures, so the discrimination of structures and textures becomes even harder. Recently, deep learning based methods [Xu et al., 2015; Liu et al., 2016; Li et al., 2016; Fan et al., 2017b; Chen et al., 2017a; Fan et al., 2017a; Shen et al., 2017] take advantage of deep neural networks that are beneficial for extracting richer image features.




(a) Existing filters (b) Our filter

Figure 1.4: Guidance in image smoothing. Existing filters only use a single structure guidance, while our filter simultaneously employs independently-generated structure guidance and texture guidance.

However, these deep models have to use the output of hand-crafted filters as ground truth for training. Therefore, they are limited by the shortcomings of these filters, and cannot learn how to appropriately distinguish structures and textures.

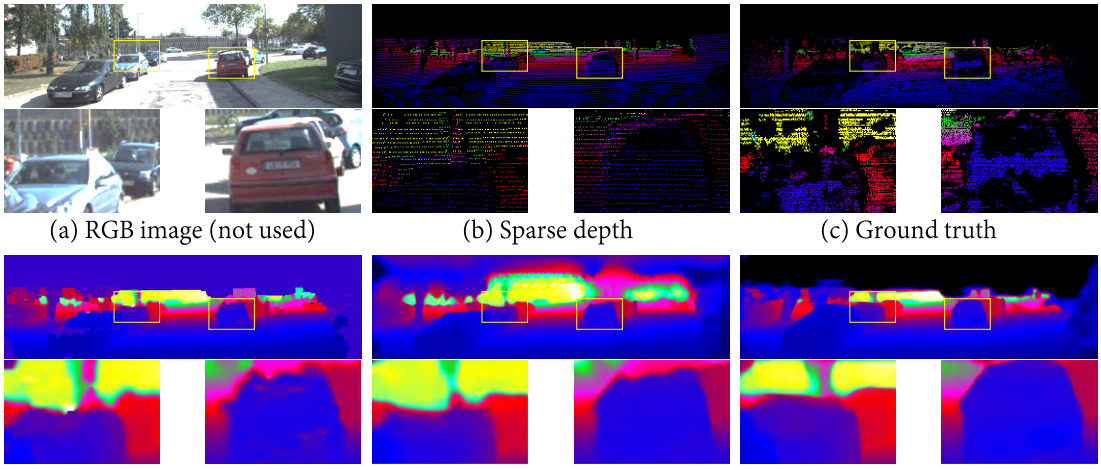
### 1.1.2 Our Motivation

Motivated by the aforementioned challenges in differentiating between structures and textures, we propose to use independent structure guidance and texture guidance. Structure guidance should indicate semantically meaningful object structures regardless of their gradients. This can be achieved by making use of existing semantic edge detection methods [Hallman and Fowlkes, 2015; Xie and Tu, 2015]. To lower the possibility of treating strong textures as edges, we introduce the concept of "texture-awareness" as a primary novelty. We realize it by using texture guidance to indicate (1) the texture region (where the texture is), and (2) the texture magnitude (strong textures have larger magnitude). Although the proposed texture guidance does not directly supply structural cues like structure guidance does, it provides more effective discrimination of textures. This is complementary to the role of structure guidance. Combining the two forms of guidance is beneficial for better identifying and removing textures without blurring structures.

### 1.1.3 Our Contributions

In this thesis, we introduce the new concept of "texture-awareness" that indicates the position and magnitude of textures. We also give theoretical insights on the relationship between "structure-awareness" and "texture-awareness", which are independent but complementary to each other. Based on this, we propose two novel image smoothing methods that simultaneously leverage structure guidance and texture guidance (see Fig. 1.4):

1. **Kernel-based double-guided filter (DGF):** DGF is the first kernel filter that incorporates both structure guidance and texture guidance into local filtering. It is free from the negative impact of gradients (or intensity difference), and able to better differentiate between structures and textures than existing hand-crafted methods. This work has been published in DICTA 2017 [Lu et al., 2017].



(a) RGB image (not used) (b) Sparse depth (c) Ground truth

(d) SparseConvs (e) S2D (f) Ours (IR)

Figure 1.5: Depth completion from sparse depth. Without the image as guidance, existing depth-only models, *e.g.*, SparseConvs [Uhrig et al., 2017] and S2D [Ma et al., 2019], present severe structure degradation, *i.e.*, inappropriately recovering semantically consistent object boundaries (*e.g.*, the car) and small/thin objects (*e.g.*, the pole). Our supervised model (IR) outperforms SparseConvs and S2D in better reducing structure degradation. All the depth maps are colorized for better visualization.

2. **Texture and structure aware deep filtering network (TSAFN):** TSAFN is the first deep filter that learns to robustly predict natural textures and combines the learned texture guidance and structure guidance to facilitate image smoothing. We also present synthetic data to enable the training of texture prediction and image smoothing. This work has been published in ECCV 2018 [Lu et al., 2018a].

Experimental results demonstrate that using the two forms of guidance enables to remove strong textures without degrading main structures, which significantly improves image smoothing performance.

## 1.2 Depth Completion

### 1.2.1 Background

Dense and accurate depth is beneficial to many computer vision tasks, *e.g.*, 3D object detection [Chen et al., 2016b; Wang et al., 2019a], optical flow estimation [Ranjan et al., 2019; Zhu et al., 2019a], and semantic segmentation [Ye et al., 2019; Zhang et al., 2019b]. However, depth maps acquired from sensors, like LiDAR, are of low quality as they are too sparse to fulfill some practical needs (see Fig. 1.5(b)). Depth completion, an enhancement technique for incomplete depth [Chen et al., 2012; Gu et al., 2017], thus aims to recover dense depth from sparse measurements.

Deep learning based depth completion models [Ma et al., 2019; Qiu et al., 2019; Eldesokey et al., 2019] have shown superior performance over traditional non-learning

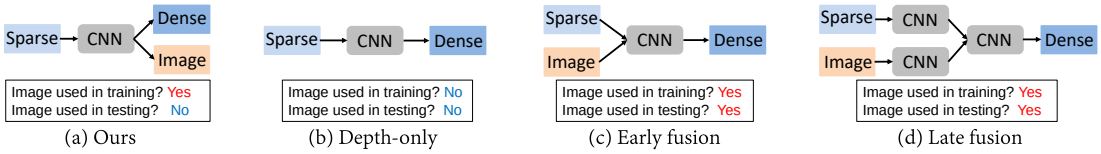
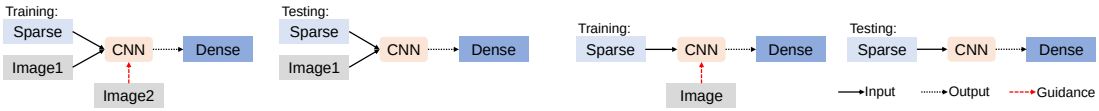



Figure 1.6: Supervised depth completion models. (a) Our model, *i.e.*, depth completion via image reconstruction (IR), takes sparse depth as the only input, and outputs a reconstructed image and dense depth simultaneously. Image reconstruction is only used as an auxiliary task at the training stage. During testing, no image is required. (b) Depth-only models input sparse depth and output the dense map. (c)-(d) Multiple-input models take the image as an additional input with an early or late fusion strategy, and the image is required in both training and testing.

methods [Kopf et al., 2007; Silberman et al., 2012; Barron and Poole, 2016; Ferstl et al., 2013] that largely rely on hand-crafted features. Hence, in this thesis, we mainly focus on learning based methods. Given dense depth ground truth available, existing studies for supervised depth completion are generally classified into *depth-only* and *multiple-input* methods. Depth-only (self-guided) methods use sparse depth as the only input [Uhrig et al., 2017; Ma et al., 2019; Eldesokey et al., 2019], as shown in Fig. 1.6(b). However, they may fail to recover semantically consistent boundaries, or full structures of small/thin objects due to the high sparsity of input depth points (see Fig. 1.5(d) and (e)). This problem, known as structure degradation in depth, can be partly resolved by using dense depth ground truth. However, most ground truth depth is not purely dense (see Fig. 1.5(c)). For example, on the KITTI Depth Completion Benchmark [Uhrig et al., 2017], the ground truth is generated by accumulating LiDAR measurements from adjacent frames with manually-removed outliers (usually existing around occluded boundaries), and it only accounts for around 30% of the image domain. Therefore, the ground truth cannot supervise every pixel, especially in important regions like boundaries. In that case, structure degradation can be further addressed by treating the RGB image as a reference-guidance. The basic assumption is that depth structures coincide with image edges [Schneider et al., 2016] and the image can provide more dense and consistent structural cues. To this end, multiple-input (reference-guided) methods take the RGB image as an additional input and incorporate image features through early or late fusion [Qiu et al., 2019; Van Gansbeke et al., 2019; Cheng et al., 2018; Jaritz et al., 2018], as illustrated in Fig. 1.6(c) and (d). Nevertheless, aggregating features from two modalities is challenging and complicated [Eldesokey et al., 2019; Qiu et al., 2019]. Also, calibrating images to depth maps can be expensive in practice [Henry et al., 2012; Kerl et al., 2015]. Further, for end-use systems such as autonomous vehicles, incorporating additional calibrated sensors and associated processing modules may significantly increase the cost.

Another practical concern is that purely-dense and high-quality depth ground truth is hard and expensive to obtain [Uhrig et al., 2017; Ma et al., 2019]. The intuitive solution is to train the model without ground truth, *i.e.*, unsupervised. For




(a) Existing models (b) Our model

Figure 1.7: Unsupervised depth completion models. (a) Existing models, *e.g.*, S2D [Ma et al., 2019] and DDP [Yang et al., 2019], take the image as an additional input in both training and test phases. A second stereo image constructs the image warping loss, which gives implicit supervision for dense depth. (b) Our model, *i.e.*, the unsupervised depth completion auto-encoder (UDAE), only uses a single image for training. At test time, we recover dense depth only from the sparse input.

unsupervised depth completion, structure degradation becomes even harder to overcome because the input itself is used as a supervision signal and it is too sparse to contain appropriate structure information [Zhang et al., 2019a; Wong et al., 2020]. Existing unsupervised works have to take the RGB image as an additional input and calculate the image warping loss either from stereo images [Yang et al., 2019] or adjacent video frames [Ma et al., 2019; Wong et al., 2020, 2021a] during training (see Fig. 1.7(a)). Clearly, compared with supervised methods where plain early and late fusion strategies are readily available, there are far fewer options for integrating image features in the unsupervised community. Moreover, the high dependence on RGB images limits the efficiency of these models in real-world applications.

### 1.2.2 Our Motivation

Motivated by the structure degradation issue in depth-only models and the practical concern of using RGB images, we continue the depth-only paradigm and aim to inject more image features so as to provide richer structural cues to reduce structure degradation in supervised and unsupervised settings. Moreover, in both frameworks, sparse depth is the only input and the image is only employed during training. At test time, we do not need any extra information other than the sparse input.

### 1.2.3 Our Contributions

In this thesis, we exploit a new approach to integrating image features in depth completion, *i.e.*, only resorting to it during training time by incorporating it as part of the training loss rather than taking it as an extra input. Based on this, we propose two novel depth completion models that employ the image as guidance only in training:

1. **Supervised depth completion via auxiliary image reconstruction (IR):** IR recovers dense depth and reconstructs the image from the sparse input simultaneously (see Fig. 1.6(a)), where dense depth ground truth and the image are used to supervise depth completion and image reconstruction respectively. It enables our model to acquire more image features from the image reconstruction branch, and thus improve the overall performance. This work has been published in CVPR 2020 [Lu et al., 2020].

2. **Unsupervised depth completion auto-encoder (UDAE):** UDAE is formulated as an auto-encoder, where sparse depth is first transformed into latent features and then recovered into dense depth (see Fig. 1.7(b)). We employ the image to guide latent features to inject more structural cues. This work has been published in WACV Workshops [Lu et al., 2022].

Experimental results show that using image guidance in training significantly improves depth completion performance in both supervised and unsupervised settings only with sparse depth as input. Note that in both models, the image is only required at the training stage, and no image is used during test time. Hence, the proposed method is more practical and deployable in real-world applications where a jointly calibrated camera is not available at run-time.

## 1.3 Thesis Outline

The remainder of the thesis is organized as follows.

**Chapter 2** In this chapter, we review existing literature on image smoothing and depth completion. Through an extensive literature review, we can have a comprehensive understanding of the two tasks, including their history, mainstream methods, and up-to-date progress.

**Chapter 3** In this chapter, we propose a novel kernel-based double-guided filter (DGF). To enhance the identification of textures, for the first time, we introduce the concept of "texture guidance" to indicate the position and magnitude of textures. Additionally, we adopt semantic edge detection as structure guidance, which is beneficial for preserving more semantically meaningful structures. The proposed DGF incorporates the two forms of guidance into the kernel operation to be both "structure-aware" and "texture-aware". Through extensive experiments, we provide the appropriate usage of the DGF and demonstrate that it can effectively remove strong textures without blurring main structures. The content in this chapter is based on our published paper [Lu et al., 2017].

**Chapter 4** In this chapter, we aim to employ deep neural networks to make the filter more adaptive to various types of textures than hand-crafted methods. To this end, we generate synthetic data by blending natural textures with clean structure-only images. With the data, we build a texture prediction network (TPN) that indicates the location and magnitude of textures, *i.e.*, texture guidance. We additionally take advantage of a semantic structure prediction network (SPN) to generate structure guidance. We then incorporate the two forms of guidance into the filtering network that constitutes our texture and structure aware filtering network (TSAFN). TSAFN is able to more effectively identify the textures to remove ("texture-awareness") and the structures to preserve ("structure-awareness"). Experimental results demonstrate that the proposed model achieves superior performance in image smoothing, and generalizes well to natural images. The content in this chapter is based on our published paper [Lu et al., 2018a].

**Chapter 5** In this chapter, we introduce a novel supervised depth completion model. The unique design is that it simultaneously outputs a reconstructed image and a dense depth map. Specifically, we formulate image reconstruction from sparse depth as an auxiliary task during training that is supervised by the unlabelled image. During testing, our system accepts sparse depth as the only input, *i.e.*, the image is not required. Our design enables the depth completion network to learn complementary image features that help to better understand object structures. The extra supervision incurred by image reconstruction is minimal, because no annotations other than the image are needed. We evaluate our method on the KITTI Depth Completion Benchmark [Uhrig et al., 2017] and show that depth completion can be significantly improved via auxiliary image reconstruction. Our algorithm consistently outperforms depth-only methods and is also suitable for indoor scenes. The content in this chapter is based on our published paper [Lu et al., 2020].
**Chapter 6** In this chapter, we focus on a more challenging task, *i.e.*, unsupervised depth completion only from sparse depth. Instead of resorting to the image as input and a second image for training like existing works, we propose to employ a single image to guide the learning process. This idea is inspired by the image guidance approach in the last chapter, but is more specific to the unsupervised setting. Specifically, we regard dense depth as a reconstructed result of the sparse input, and formulate our model as an auto-encoder. To reduce structure degradation resulting from sparse depth, we employ the image to guide latent features by penalizing their difference in the training process. The image guidance loss enables our model to acquire more dense and structural cues that are beneficial for producing more accurate and consistent depth values. For inference, our model only takes sparse depth as input and no image is required. Our paradigm is new and pushes unsupervised depth completion further than existing works that require the image at test time. On the KITTI Depth Completion Benchmark [Uhrig et al., 2017], we validate its effectiveness through extensive experiments and achieve good performance compared with other unsupervised works. The proposed method is also applicable to indoor scenes. The content in this chapter is based on our published paper [Lu et al., 2022].
**Chapter 7** In this chapter, we summarize main contributions of the thesis, and propose potential future work for both image smoothing and depth completion.

## 1.4 Publications

The contributions in this thesis are based on the following four published papers.

1. **Kaiyue Lu**, Nick Barnes, Saeed Anwar, Liang Zheng. "From Depth What Can You See? Depth Completion via Auxiliary Image Reconstruction". *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), 2020.*

2. **Kaiyue Lu**, Shaodi You, Nick Barnes. "Deep Texture and Structure Aware Filtering Network for Image Smoothing". *Proceedings of the European Conference on Computer Vision (ECCV), 2018.*

3. **Kaiyue Lu**, Nick Barnes, Saeed Anwar, Liang Zheng. "Unsupervised Depth Completion Auto-Encoder". *Proceedings of the Winter Conference on Applications of Computer Vision (WACV) Workshops, 2022.*

4. **Kaiyue Lu**, Shaodi You, Nick Barnes. "Double-Guided Filtering: Image Smoothing with Structure and Texture Guidance". *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2017.*

## 1.5 Summary

In this chapter, we have given a high-level review on image enhancement, which is a typical task in image processing. We have found that structure degradation widely exists in image enhancement. To address this issue, we have focused on two specific tasks, *i.e.*, image smoothing and depth completion, and illustrated our motivation on employing guidance to improve the overall performance. We have summarized main contributions in the thesis. We have also overviewed the thesis outline and listed our publications. From next chapter, we will first review existing literature on the two tasks, and then detail our research works.

Chapter 2

# Literature Review

## 2.1 Image Smoothing

Given an image $I$, the goal of image smoothing is to remove insignificant textures in the image and generate a smoothed output $\tilde{I}$ that contains main structures.

### 2.1.1 Kernel Methods

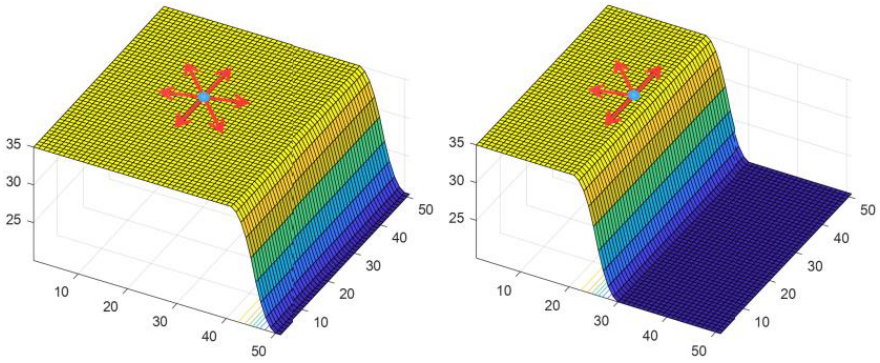
The target pixel $p$ is located at the centre of the squared kernel (kernel-centred), and its new value is a weighted average of pixels in its neighborhood $\Omega$:

$$\tilde{I}_p = \frac{1}{\kappa_p} \sum_{q \in \Omega} w_{pq} I_q, \tag{2.1}$$

where $w_{pq}$ is the weight between $p$ and its neighbor $q$ and $\kappa_p = \sum_{q \in \Omega} w_{pq}$ is used for normalization. For kernel methods, $w_{pq}$ is the most essential coefficient to be designed and adjusted. The key principle is that only pixels with similar color features to the kernel-centred one should be smoothed together [Shang et al., 2021]. Based on this, we illustrate two typical cases in Fig. 2.1: (a) Textured regions, where smoothing should involve all the local pixels to suppress textures; (b) Edge regions, where only pixels on the same side of the edge should be involved to prevent edge blurriness. The weight should be adaptive to different region characteristics in order to effectively differentiate between structures and textures. Moreover, this weight is calculated either from the input itself, *i.e.*, self-guided, or another guidance image, *i.e.*, reference-guided. We review the two categories of kernel methods below.

#### 2.1.1.1 Self-guided methods

Color/intensity difference is the mostly widely-used quantity in kernel methods to identify edges, *i.e.*, the difference is normally large across edges. The bilateral filter is one typical filter that is dependent on color difference. For better performance, color difference is often combined with other statistics or methods, *e.g.*, region covariances, co-occurrence, superpixels, geometrically adaptive support regions, and edge maps.




(a) Texture regions (b) Edge regions

Figure 2.1: Illustration of kernel filtering in typical regions. (a) Texture regions. The filter should involve all the local pixels to suppress textures. (b) Edge regions. Only pixels on the same side of the edge should be involved to prevent edge blurriness. Image taken from [Shang et al., 2021].

**Bilateral filter.** The bilateral filter (BLF) was first proposed in [Tomasi and Manduchi, 1998], where the weight is composed of a spatial kernel and a range kernel, *i.e.*,

$$w_{pq}^{BLF} = \exp\left(-\frac{\|\mathbf{p}-\mathbf{q}\|^2}{2\sigma_s^2} - \frac{\left\|I_p - I_q\right\|^2}{2\sigma_r^2}\right). \tag{2.2}$$

where $\sigma_s$ and $\sigma_r$ control the sensitivity of spatial and range support respectively. Intuitively, pixels that are closer to the centre of the kernel with similar intensities are more likely to be smoothed together with larger weights. The impact of far-away pixels or those with large color difference (on the other side of the edge) should be lowered by being assigned with smaller weights. However, the BLF has three major disadvantages: (1) Inefficiency. The combination of the two Gaussian kernels makes the BLF suffer from high computational costs. Several works attempt to accelerate the BLF [Paris and Durand, 2006; Yang et al., 2009; Gastal and Oliveira, 2012; Porikli, 2008; Durand and Dorsey, 2002; Yang et al., 2009, 2015]. (2) Lack of robustness to strong textures. Only using color difference to differentiate between structures and textures is not robust because some strong textures may also present large color contrast. In that case, structures may be blurred when the filter tries to remove strong textures. (3) Gradient reversal. A small $\sigma_r$ may lead to over-sharpened edges in the smoothed image, so that they are reversely amplified in detail enhancement [He et al., 2013; Bae et al., 2006; Durand and Dorsey, 2002]. The latter two issues can be improved by incorporating the edge sharpness term [Khetkeeree and Thanakitivirul, 2020], constraining the range kernel [Yang, 2015], leveraging adaptive kernel sizes [Ghosh et al., 2019], or making use of the inherent and discriminative properties of structures and/or textures (introduced below).

**Region covariances.** Region covariances, a kind of region descriptor, are second-order statistics which are robust to illumination changes [Tuzel et al., 2006]. Karacan et al. [2013] observe that patches with similar structure or texture patterns tend to

have similar covariances, so they can be used to measure the local similarity of pixels. The feature representation of each pixel consists of 7 components, *i.e.*, intensity, first and second order derivatives along x and y directions, and coordinates. Assisting the intensity with other statistics is beneficial for better grouping similar image patterns and achieving more robust smoothing performance.

**Co-occurrence.** Textures are normally regarded as repetitive patterns in the image regardless of their intensities [Cai and Baciu, 2012]. Co-occurrence information, originally used to describe textures [Haralick et al., 1973], is useful in measuring repetitive pixel values. The co-occurrence filter (COF) [Jevnisek and Avidan, 2017] is thus designed to identify textural regions. Specifically, the COF replaces the range kernel in the BLF with a normalized co-occurrence matrix. This new weight becomes larger when neighbouring pixel values co-occur more frequently, *i.e.*, texture regions. In that case, those pixels can be smoothed together. For structures, the co-occurrence is less so that the weight is smaller to avoid edge blurriness. Users can freely choose the region to calculate the co-occurrence, *e.g.*, the entire image or local patches.

**Superpixels.** Superpixels divide the image into several non-overlapping regions that have similar intensities and adhere to boundaries [Achanta et al., 2012]. The segment graph filter (SGF) [Zhang et al., 2015] takes advantage of superpixels and makes the filter edge-aware by introducing a weight calculated from sliding a window towards adjacent superpixels. Li et al. [2018] empirically observe that hard superpixels, *e.g.*, SLIC [Achanta et al., 2012] used in the SGF, may result in artifacts around boundaries. To deal with this issue, they alternatively employ a soft clustering algorithm [Adams et al., 2010] that enables gradual changes along boundaries. By iteratively performing soft clustering, smoothing results can be significantly improved.

**Geometrically adaptive support regions.** Most kernel filters assume that the kernel is a fixed box which largely limits its ability to keep consistent with object boundaries. Even though superpixels are more structure-friendly, they only describe local regions that cannot reflect full object structures. To realize arbitrary shapes for support regions, the cross-based local multipoint filter (CLMF) [Lu et al., 2012] first selects certain points from a series of candidates within a shape-adaptive local region, and then aggregates several multipoint estimates vertically and horizontally (or the reverse order). However, the support region of the CLMF is largely dependent on the order of aggregation, and the complexity may be increased due to the long arms. The local polynomial approximation based multipoint filter (MLPA) [Tan et al., 2014] is proposed to improve the CLMF by using a 2D quadratic spatial regularization term and expanding support regions in both horizontal and vertical directions. The tree filter (TF) [Bao et al., 2014] enables a longer distance propagation through the Minimum Spanning Tree (MST) [Frieze, 1985] but with larger computational costs. Dai et al. [2015] propose the fully connected guided filter (FCGF) to take advantage of MST [Frieze, 1985] to estimate the support region from the entire image, and apply the local multipoint filtering framework [Katkovnik et al., 2010] to achieve linear-time filtering. The FCGF is able to acquire more relevant pixels for smoothing.

**Edge maps.** Edge maps provide direct structure information. The key point is how to generate proper edge guidance and incorporate it into the filter. Shang et al.

[2021] utilize the classical Canny operator [Canny, 1986] for edge detection, and apply it as a constraint to the Gaussian kernel. However, Canny edges always contain noise, which causes confusion between edges and textures. To acquire more appropriate edges, Cho et al. [2014] propose modified Relative Total Variation (mRTV) based on gradients within a local patch to infer edges. Zang et al. [2015] summarize three essential properties of edges, *i.e.*, anisotropy (intensity variations in different orientations), non-periodicity (main structures have less oscillation of intensity variation while textures have more), and local directionality (the intensity variation of main structures has more consistent local directionality than textures). Based on these, they design the directional anisotropic structure measurement (DASM), which effectively identifies structures and is robust to color contrast and object scales. Note that the three edge methods above are hand-crafted, so they are not close to human perception due to the lack of semantic meaning. To deal with this issue, Yang [2016] makes use of the semantic edge detection model [Dollár and Zitnick, 2015] trained on human-labelled data, and incorporates the semantic edge map as a confidence value to constrain intensity difference. This approach can produce more human-pleasing smoothing results, especially in preserving semantic boundaries.

#### 2.1.1.2 Reference-guided methods

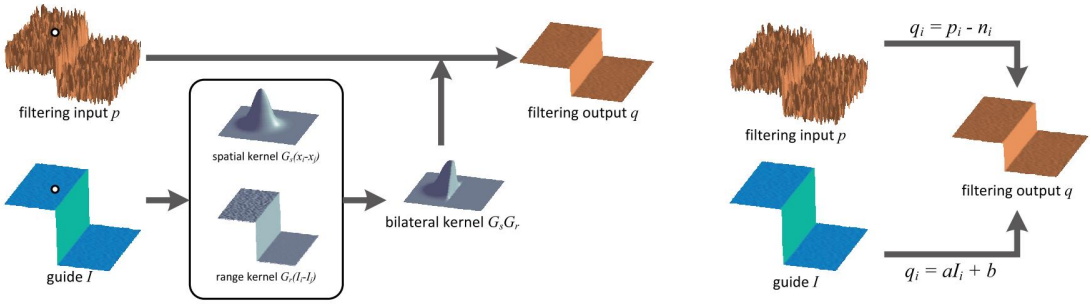
In many cases, the single image alone cannot provide sufficient structure information for smoothing. Hence, another image, named the *guidance image*, is employed to supply more structural cues. In image smoothing, the guidance may come from different conditions, *e.g.*, flash and no-flash [Petschnigg et al., 2004], daytime and night [Guo et al., 2020], or different modalities, *e.g.*, depth [Kopf et al., 2007; Lo et al., 2017], NIR [Yan et al., 2013; Sharma et al., 2017].

**Joint bilateral filter.** The joint bilateral filter (JBF) [Petschnigg et al., 2004; Eisemann and Durand, 2004] extends the BLF by computing the range weight from the guidance image $I^{Ref}$ (see Fig. 2.2(a)). That is,

$$w_{pq}^{JBF} = \exp\left(-\frac{\|\mathbf{p}-\mathbf{q}\|^2}{2\sigma_s^2} - \frac{\left\|I_p^{Ref} - I_q^{Ref}\right\|^2}{2\sigma_r^2}\right). \tag{2.3}$$

Although the JBF is equipped with more structure information, the guidance is static, *i.e.*, fixed in all iterations. In that case, with the progress of smoothing, some irrelevant content, *e.g.*, textures, in the original guidance image may be continuously transferred into the smoothed image.

**Rolling guidance filter.** Considering the static nature of the guidance image in the JBF, the rolling guidance filter (RGF) [Zhang et al., 2014b] is proposed to update the guidance image iteratively. Specifically, the RGF is composed of two steps, *i.e.*, small structure removal and edge recovery. Small structure removal is achieved by applying the Gaussian filter with different scales to remove most details. However, this operation inevitably blurs structures as the Gaussian filter does not consider any intensity cues. Afterwards, the blurred image is used as guidance to smooth



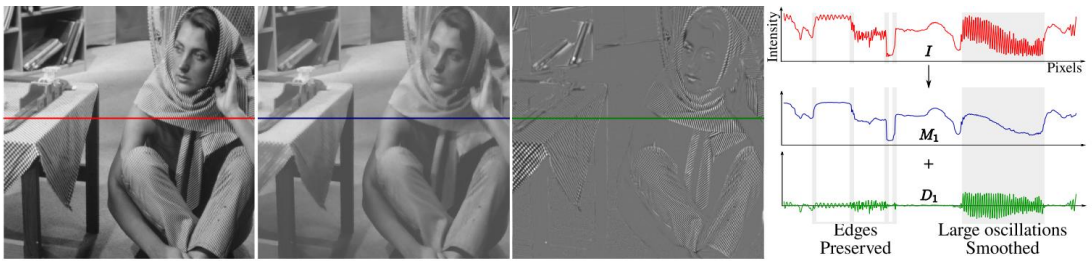

(a) Joint bilateral filter (b) Guided filter

Figure 2.2: Illustration of reference-guided kernel filters. (a) The joint bilateral filter [Petschnigg et al., 2004; Eisemann and Durand, 2004] calculates the range kernel from the guidance image. (b) The guided filter (GF) [He et al., 2013] assumes the target image can be linearly transformed from the guidance image. Image taken from [He et al., 2013].

the original input, and then the guidance image is replaced with the new smoothed output. By this means, structures can be gradually recovered without introducing unwanted details. Although the guidance image is obtained from the input itself, we still categorize the RGF as a reference-guided method because the guidance image is obtained from different conditions, *i.e.*, the blurred version.

**Guided filter.** When comparing the RGF and the JBF, we realize the importance of accurately capturing the structure dependency/consistency between the input image and its guidance [Liu et al., 2017c]. The guided filter (GF) [He et al., 2013] gives new insight on this problem by assuming a linear relationship between the two, *i.e.*, the input image can be linearly transformed from its guidance (see Fig. 2.2(b)). Compared with the BLF (or the JBF, depending on whether the guidance is the input itself or another image), the GF has three major advantages: (1) Better smoothing performance. The GF can preserve edges more effectively without introducing the gradient reversal artifact. (2) More efficiency. The time complexity of BLF is $O(Nr^2)$, where $N$ is the number of pixels and $r$ is the kernel radius. By contrast, the GF is an $O(N)$ approach. Hence, the GF is more efficient, and the complexity is not affected by the kernel radius. (3) Broader applications. The GF can be more broadly applied to various applications with higher quality, *e.g.*, HDR compression, image dehazing, image matting.

However, the GF still suffers from two problems: (1) halo artifacts (blurred edges due to their low color contrast or being close to strong textures), and (2) the fixed box window (no arbitrary kernel sizes adaptive to image content and the weight of each pixel is 1). To reduce halo artifacts, the edge-aware weighting scheme is widely adopted, which is based on normalized local variances of pixels [Li et al., 2014c,b], gradients [Kou et al., 2015], edge direction via the steering kernel regression [Sun et al., 2019], or coefficient propagation [Mun et al., 2018]. To enable arbitrary kernel sizes, Fukushima et al. [2018] make use of the binary weighting function to assign different weights to neighbouring pixels. This operation is implicit because it does




(a) Input (b) Structure layer (c) Texture layer (d) Scanline-plots

Figure 2.3: Illustration of structure and texture layer decomposition in global image smoothing methods. Image taken from [Subr et al., 2009].

not change the spatial size but the weight values.

### 2.1.2 Global Methods

Global methods assume the image can be decomposed into a structure layer (smoothed output) and a texture layer (always discarded), as shown in Fig. 2.3. They achieve the decomposition by solving a globally-defined objective function, *i.e.*,

$$\tilde{I}^* = \arg\min_{\tilde{I}} \sum_p (I_p - \tilde{I}_p)^2 + \lambda \cdot R(\tilde{I}), \tag{2.4}$$

where $(I_p - \tilde{I}_p)^2$ is the data term that maintains the similarity between the input and output, and $R(\tilde{I})$ is the regularization term that contains image priors and regulates the smoothness of the output. $\lambda$ controls the impact of the regularizer and is non-negative. Compared with kernel filters, global methods are usually better at suppressing textures because they are performed on the entire image domain and can process textures in a global view. The limitation is the lack of efficiency resulting from optimizing the non-convex objective function [Pan et al., 2019; Liu et al., 2020d]. Considering that the priors are either from the input itself or the guidance image, in the following, we review global algorithms in terms of the property of guidance, *i.e.*, self-guided and reference-guided.

#### 2.1.2.1 Self-guided methods

Gradients defined in Eq. 1.2 are widely adopted as part of the regularizer because structures normally have large gradients while textures are just small oscillations in intensity. In self-guided methods, the regularizer manipulates different levels of gradients so that they are gradually smoothed from small to large. However, similar to color difference in kernel methods, gradients are not robust enough to identify strong textures. Some other constraints are also frequently employed in the regularization process, *e.g.*, local extrema, co-occurrence, edge maps, and combination with kernel filters (this is mainly for accelerating global optimization).

**Gradients.** The pioneering work, Total Variation (TV) [Rudin et al., 1992], directly

penalizes the $\ell_1$ norm of gradients. This prior has been widely used in other global methods [Yin et al., 2005; Aujol et al., 2006; Dou et al., 2017]. However, as discussed above, simply relying on gradients may not accurately differentiate between structures and textures. To improve it, Relative Total Variation (RTV) [Xu et al., 2012] is proposed to consider their different properties, *i.e.*, the direction of gradients is more similar in local edges than textures. Based on this, RTV computes total variations in a box window by simultaneously including the magnitude and direction of gradients. Zhao et al. [2019] extend RTV by individually processing each channel to obtain color-sharing and color-discriminative information. However, RTV suffers from a lack of robustness to different scales of textures. $\ell_0$ smoothing [Xu et al., 2011], which constrains the number of non-zero gradients, is more adaptive to scales because the optimization process involves a discrete metric that is free of scales. The extension of $\ell_0$ smoothing includes using the truncated operation to prevent the algorithm from penalizing large gradients [He and Wang, 2018], making the parameter choice more adaptive [Ni and Wu, 2018], improving the robustness to high-contrast textures [Fang et al., 2019a], and accelerating the optimization process [Nguyen and Brown, 2015], combining it with the $\ell_1$ fidelity [Shen et al., 2012]. Although the $\ell_2$ norm is also used in optimization [Farbman et al., 2008; Liu et al., 2013b; Min et al., 2014; Liu et al., 2017b], its performance is generally behind $\ell_1$ and $\ell_0$ with more halo artifacts introduced [Shen et al., 2012]. In summary, constraining gradients is popular in global smoothing, but how to distinguish structures and textures more effectively based on gradients is still under study.

**Local extrema.** For the purpose of reducing the impact of color contrast in distinguishing structures and textures, Subr et al. [2009] propose to use the local extrema, *i.e.*, maxima and minima in a local region, for characterization. Specifically, structures are regarded as large variations in intensities of local neighbouring extrema, while textures are oscillations between local maxima and minima. The advantage of this assumption is that it is not affected by contrast, *i.e.*, even weak structures and strong textures can be identified. Minimal and maximal extremal envelopes are computed by interpolating local intensities through optimization. The smoothed values are the mean between the two envelopes. This method gives new insight into structure-texture discrimination, but the overall performance is largely restricted by the quality of local extrema, *e.g.*, they may be affected by the size of the local region or the mixture of structures and textures.

**Co-occurrence.** The co-occurrence can describe the repetitive property of textures, and has been applied in the kernel filter [Jevnisek and Avidan, 2017] (see Section 2.1.1.1). However, two problems may exist: (1) The co-occurrence may not be intuitive in weakly-periodic or highly-random textures; (2) Intensities along edges may also present co-occurrence, which is likely to cause confusion with textures. To overcome the two issues, Xu et al. [2021] exploit a discriminative prior on patch co-occurrence, *i.e.*, recurrent patches along edges tend to have a major direction while those of textures just scatter around. Hence, structures and textures can be distinguished via the spatial distribution of recurrent patches. This prior is incorporated into the Morphology Component Analysis (MCA) framework [Starck et al., 2005],

which largely reduces the ambiguity in differentiating between structures and textures.

**Edge maps.** Edge maps intuitively supply structure information. Liu et al. [2020e] find that some weak edges with small gradients are prone to be over-smoothed, so they first use the Canny operator [Canny, 1986] to generate edge maps and then enhance those regions containing edges with histogram equalization. Afterwards, they perform $\ell_0$ smoothing to enhanced images for better structure preservation. Guo et al. [2018] slightly modify the Canny operator and propose to refine edge maps simultaneously with image smoothing. The edges in these two works are hand-crafted, so they may be not close to human perception. To inject more semantic information, Li et al. [2017] incorporate (1) the edge potential, *i.e.*, edge confidence from [Dollár and Zitnick, 2015] to measure the boundary strength between two pixels, and (2) the semantic potential, *i.e.*, semantic labels from [Krähenbühl and Koltun, 2011] to enforce the consistency of two pixels that belong to the same semantic category, into the regularization term. Zhu et al. [2018] jointly optimize saliency object detection, semantic edge detection [Xie and Tu, 2015], and image smoothing, which is beneficial for preserving more semantically meaningful structures.

**Combination with kernel filters.** Generally, global methods lack efficiency due to the iterative optimization process while kernel methods run faster but may more easily suffer from heavy halo or gradient reversal artifacts. Combining the two can achieve both desirable efficiency and smoothing quality, as demonstrated in [Barron and Poole, 2016; Liu et al., 2018a] (both works embed the bilateral filter [Tomasi and Manduchi, 1998] to WLS [Farbman et al., 2008]).

#### 2.1.2.2 Reference-guided methods

The guidance image offers additional structural cues to the original input. However, most works ignore that fact that there may exist structure inconsistency between the two images, *i.e.*, edges appear in one image but not in the other image [Shen et al., 2015]. In that case, some irrelevant or even erroneous structures may be transferred to the target image and thus degrades the smoothing performance. Reference-guided global methods aim to deal with the structure inconsistency issue.

**Mutual structure filter.** The goal of the mutual structure filter (MSF) [Shen et al., 2015] is to capture the common structure information (named as *mutual structures*) between the target and guidance images while suppressing or discarding uncommon structures (named *inconsistent structures*). To this end, normalized cross correlation (NCC) based on gradients is employed to measure the structure similarity between two image patches, *i.e.*, a large NCC (*e.g.*, close to 1) indicates the two patches have common edges. With this constraint, the negative impact of inconsistent structures can be reduced. Moreover, the guidance image is jointly optimized with the target one, which is beneficial for obtaining more accurate structures from guidance [Perona and Malik, 1990; Zhang et al., 2014b].

**Static and dynamic filter.** The static and dynamic filter (SDF) [Ham et al., 2015, 2017] aims to capture structural dependencies and inconsistencies between the target

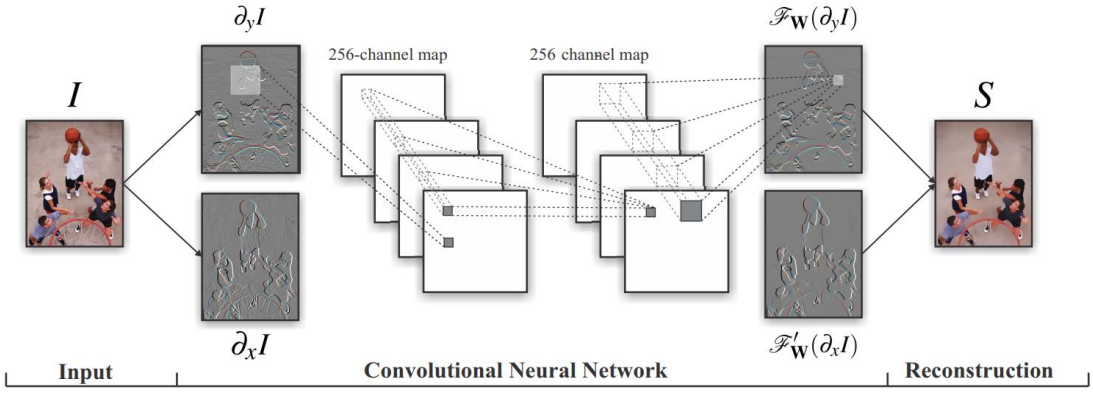



Figure 2.4: Network architecture of the deep edge-aware filter (DEAF) [Xu et al., 2015]. The network is built on the gradient domain of the image, which benefits the recovery of sharp edges. The output is reconstructed from gradients. Image taken from [Xu et al., 2015].

and guidance images more robustly. This is based on a further observation that both the initial guidance image (named as *static guidance*) and the smoothed image (named as *dynamic guidance*) are useful to image smoothing. Specifically, static guidance contains the original and rich structure information (but also some irrelevant cues). Structures in dynamic guidance are more appropriate, but some of them may be lost after several iterations (see the analysis of the Rolling Guidance Filter in Section 2.1.1.2). Hence, the SDF adaptively acquires structure information from both static and dynamic guidance images, which is achieved by incorporating static guidance as a constraint to the gradients of dynamic guidance.

**Mutually guided filter.** The mutually guided filter (muGIF) [Guo et al., 2017, 2020] further improves the MSF and SDF by considering three modes, *i.e.*, static/dynamic guidance, dynamic/dynamic guidance, and only dynamic guidance. The distinction is that the pixel-level measurement of structure similarity using the newly-defined *relative structure*. This filter is a more general framework in reference-guided methods, and achieves superior smoothing results by making full use of guidance.

### 2.1.3 Deep Learning Methods

#### 2.1.3.1 Self-guided methods

The pioneering work in self-guided deep image smoothing is the deep edge-aware filter (DEAF) [Xu et al., 2015] (see Fig. 2.4). It aims to approximate existing hand-crafted filters, *i.e.*, using filtered results as ground truth. The network is built on the gradient domain of the image, which benefits the recovery of sharp edges. Compared with hand-crafted filters, the DEAF has three advantages: (1) More robust edge-awareness (it comprehensively learns the edge preservation ability from various filters); (2) Higher efficiency (the optimization and inference speed is much faster than hand-crafted approaches because of the linear complexity); (3) A more unified

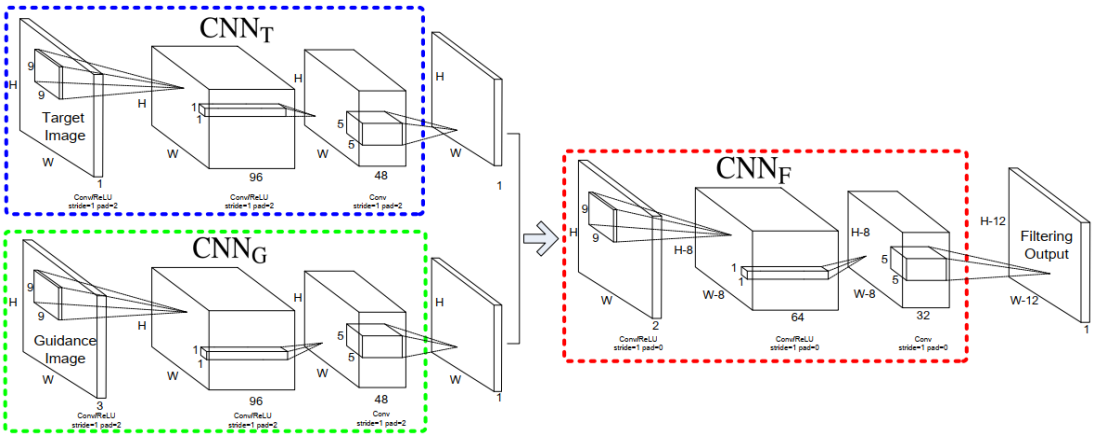



Figure 2.5: Network architecture of the deep joint image filter [Li et al., 2016]. The network is composed of three sub-networks, *i.e.*, $CNN_T$ for extracting features from the target image, $CNN_G$ for extracting features from the guidance image, and $CNN_F$ for aggregating target and guidance features. Image taken from [Li et al., 2016].

framework (the network can approximate any filter). Subsequent works follow the setting of approximating existing filters, and propose diverse network architectures. Liu et al. [2016] first use CNNs to generate an edge map from the input, and then the edge guidance is incorporated into a recurrent neural network for recurcive filtering (this method assumes the image is a group of sequences and the RNN architecture can reduce parameters and enhance the stability of network training). Similarly, Fan et al. [2017b] construct the edge network (E-CNN) to predict the edge confidence map based on local gradients, and use it to guide the filtering network (I-CNN). For better performance and faster convergence, both networks take advantage of the residual block [He et al., 2016]. The use of edge guidance can also be found in [Kim et al., 2019a]. Chen et al. [2017a] utilize the context aggregation network (CAN) [Yu and Koltun, 2015] to capture more contextual information. Wu et al. [2018] consider the guided filter [He et al., 2013] as a group of spatial varying linear transformation matrices, which can be end-to-end learned. The network absorbs the benefits of the guided filter in structure preservation and high processing efficiency. Zhu et al. [2020] design the non-local block to enhance the connection between pixels.

One common issue of these deep filters is that they have to take the output of existing filters as ground truth. Even though Zhu et al. [2019b] provide a dataset for training smoothing networks, the ground truth images are still obtained from hand-crafted filters, *i.e.*, selecting the best smoothed results by manually tuning parameters. Hence, current deep models are unable to overcome the natural deficiency of hand-crafted filters, *i.e.*, poor discrimination of structures and textures.

#### 2.1.3.2 Reference-guided methods

In deep filters, the role of the guidance image is also to provide more structural cues to the target image. Li et al. [2016, 2019] design three sub-networks for feature

extraction from the target image, feature extraction from the guidance image, and feature aggregation respectively (see Fig. 2.5). Shi et al. [2021] take advantage of the unsharp masking [Polesel et al., 2000] to enhance edges without introducing additional coefficients, which contributes to both improved results and good efficiency. The convolutional kernel in conventional CNNs is a regular grid, which may not be adaptive to spatially-variant image content. To improve it, Kim et al. [2021] propose the deformable kernel network (DKN) to adaptively learn both neighbours and corresponding weights for each pixel. This data-driven approach yields more robust smoothing results. A similar usage of the deformable network can be found in [Fang et al., 2019b]. In addition to image smoothing, these reference-guided deep filters can also applied to depth map upsampling, image colorization, noise reduction in RGB/NIR and flash/non-flash image pairs, and so on [Li et al., 2016].

#### 2.1.3.3 Unsupervised methods

As mentioned above, the ground truth for training deep filters is not reliable enough as it is derived from hand-crafted methods. However, involving human annotations for desirable ground truth is hard to achieve. Thus, unsupervised models are proposed to relax the need of resorting to pre-generated labels. Fan et al. [2018] decompose the objective function into three parts: (1) A data term to enforce the structural similarity between the input and the smoothed output; (2) An edge-preserving term to penalize important structural responses between the guidance and target images (indicated by a binary edge map); (3) A smoothness term to regulate intensity difference between neighbouring pixels. The three terms are jointly optimized by the deep network without any supervision. Zhou et al. [2019] facilitate the data term by incorporating distinctive structure and texture measures into the network. Specifically, the structure-aware measure relies on gradients (edges have larger gradients and the directions of their gradients are more uniform). Meanwhile, the texture-aware measure is obtained by searching repetitive patterns within a local neighbourhood based on HOG features [Dalal and Triggs, 2005]. This method enables better discrimination of structures and textures. Generally, in terms of the objective function, unsupervised methods are similar to hand-crafted global approaches. However, they contain more constraints and have better optimization capacity and efficiency with deep networks than hand-crafted ones.

## 2.2 Depth Completion

### 2.2.1 Non-Learning Methods

Non-learning depth completion is always associated with depth image upsampling, which improves the quality of low-resolution, noise-corrupted, or incomplete depth.

#### 2.2.1.1 Depth-only (Self-guided) methods

Non-learning depth-only methods produce a dense depth map only from the sparse input. Traditional interpolation approaches, *e.g.*, nearest, bilinear, bicubic, can naively fill in missing values by searching nearby available values, but they purely ignore the structural prior inherent in depth and may generate undesirable artifacts [Yao et al., 2020]. Hornácek et al. [2013] exploit the patch-wise self-similarity in the depth map, *e.g.*, repetition of geometric primitives and object symmetry. This helps with the recovery of more consistent depth boundaries and fine details. Ku et al. [2018] treat depth completion as a pure image processing problem, and use kernel operations, *e.g.*, dilation, closure, Gaussian blurring, to generate dense depth and enhance boundaries. This method is efficient, and can be cheaply deployed on embedded systems. Currently, there are very few works falling into the non-learning depth-only category. Without image guidance and only dependent on hand-crafted features, these approaches cannot produce high-quality depth values, especially around object boundaries.

#### 2.2.1.2 Multiple-input (Reference-guided) methods

Given the registered RGB image with depth available, multiple-input methods show superior performance over depth-only approaches because the image provides more dense and structural cues. How to effectively make use of image guidance has drawn much attention. Existing non-learning methods with image guidance generally have two categories, *i.e.*, local and global methods.

**Local methods.** Local methods take the similar form of kernel filtering, *i.e.*, filling in missing depth values within a local squared region with the target pixel located at the centred position. They basically assume that: (1) pixels with similar intensities in the RGB image are more likely to have similar depth values, and (2) large intensity changes in the image tend to correspond to depth edges. The joint bilateral filter [Haralick et al., 1973] and its variants [Riemens et al., 2009; Chen et al., 2016a; Lu et al., 2018b; Kim et al., 2014; Silberman et al., 2012] calculate the intensity/color distance from the image instead of the depth map. To involve more neighbouring pixels, the color distance can be replaced by the geodesic distance, which is defined in the 8-connected image grid [Liu et al., 2013a]. This is beneficial for generating sharper boundaries because it integrates intensity changes along geodesic curves. In addition, anisotropic diffusion treats available depth values as heat sources and diffuses depth from these sources to unknown regions [Liu and Gong, 2013; Yao et al., 2020]. Another strategy for enhancing depth edges is to first interpolate the depth input with bicubic interpolation, and then refine and sharpen depth edges with image guidance as a post-processing step [Hua et al., 2015]. Miao et al. [2012] divide depth completion into two sub-tasks, *i.e.*, smooth region inpainting and edge region inpainting. They design different algorithms for the two tasks for the purpose of improving various scales of hole filling and edge alignment. To further boost the consistency between the image and depth, Yang et al. [2014] propose an adaptive autoregressive model to minimize prediction errors. However, as Lu et al. [2014]

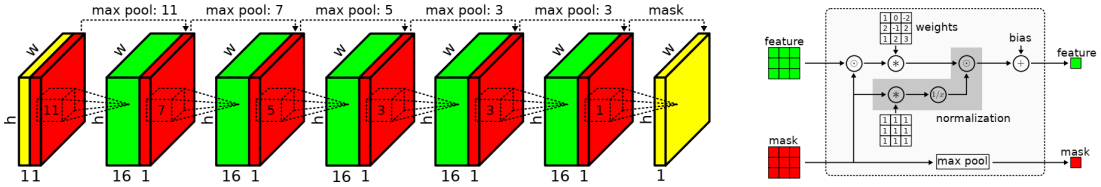
point out, depth edges cannot be well enhanced if the image is noisy or the intensity changes on edges are small, in which case the correlation between the image and depth is relatively weak. To deal with this issue, they assume similar RGB-D patches mostly exist in a low-dimensional sub-space, and impose a low-rank constraint. This method significantly strengthens the correlation between the image and depth especially in structural regions. The depth completion performance can be largely improved.

**Global methods.** Global methods assume that depth values vary smoothly across the entire depth map. They formulate depth completion as a constrained optimization problem, which consists of a data term, *i.e.*, the matching cost between predicted and known depth values, and a smoothness term, *i.e.*, penalizing edge discontinuities based on image guidance [Hawe et al., 2011]. The data term is used in almost all cases, so the research focus is to design a proper smoothness term in order to maintain consistent depth structures. Many works take advantage of Markov Random Fields (MRF) [Li, 1994] for constructing the smoothness term as well as optimization. In a standard MRF framework, the depth smoothness potential is defined as a weighted quadratic distance between neighbouring depth points, and the weighting factors connect image edge information to depth [Diebel and Thrun, 2005]. Other smoothness terms used in MRF research include bidirected image gradients and region segmentation [Kim and Yoon, 2012], edge patches [Xie et al., 2015], self-similarity of patches [Li et al., 2014a], non-local structure regularization [Park et al., 2014, 2011], and so on. Another popular method for depth smoothness is total variation (TV) [Barbero and Sra, 2011], which aims to preserve depth edges with the assumption that depth values and image intensities have a linear correlation within a small local patch [Liu et al., 2013c; Ferstl et al., 2013]. Additionally, compressive sensing [Hawe et al., 2011], adaptive region selection [Chen et al., 2013], semantic segmentation [Schneider et al., 2016], the combination of local and global information [Barron and Poole, 2016], have also been successfully applied in global optimization.

### 2.2.2 Supervised Learning Methods

#### 2.2.2.1 Depth-only (Self-guided) methods

The sparse depth input usually presents irregular depth point distribution and/or large holes (missing areas). Standard CNNs, which are designed for processing dense data, have to be aware of the sparseness of the input. The naive approach either sets invalid pixels that do not have depth values to zero [Chen et al., 2017b] or adds an extra guidance mask to the network [Köhler et al., 2014] to indicate the validity of each pixel (this mask is from the input itself so it is a self-guidance). However, it cannot effectively improve the performance of the network because invalid pixels are normally irregular and cannot be aligned and consistent with the regular pixel grid, *e.g.*, the squared kernel [Uhrig et al., 2017]. Additionally, CNNs perform poorly when the levels of sparsity significantly vary in training and test sets [Uhrig et al., 2017]. To deal with the sparsity issue, Sparsity Invariant CNNs (SparseConvs) [Uhrig et al., 2017] are proposed as a pioneering work for sparsity awareness, as illustrated




(a) Network architecture (b) Sparse convolution

Figure 2.6: Network architecture of SparseConvs [Uhrig et al., 2017]. (a) A sparse convolution is incorporated into the standard convolutional layer to indicate the validness of depth points (1 for points that have depth values and 0 for none). (b) Detailed architecture of the sparse convolution. Image taken from [Uhrig et al., 2017].

in Fig. 2.6. The motivation is to make the network output invariant of various levels of sparsity, *i.e.*, keeping robustness to different numbers of missing points. This is achieved by defining a binary mask (1 for valid pixels and 0 for invalid) for only evaluating valid pixels during network propagation. This mask is updated and propagated with the max pooling operation to indicate remaining locations of incomplete data. Within a local region, a pixel is identified as "unobserved" if all of its neighbouring values are missing. In that case, the sparsity can be explicitly modelled in the training procedure. It also enables the network to be more robust to sparsity at test time. It has been proved that SparseConvs outperform standard CNNs with more accurate depth completion results as well as faster and smoother network training.

SparseConvs still have three primary limitations: (1) It gradually downsamples feature maps and does not effectively integrate low-level and high-level features [Huang et al., 2019]; (2) It is easy to produce blurry edges [Jaritz et al., 2018]; (3) The validity mask is prone to be saturated in early layers so the validity information is quickly lost in later layers [Jaritz et al., 2018]. The first issue is related to the network architecture, so HMS-Net [Huang et al., 2019] handles it by making use of a encoder-decoder network and integrating multi-scale features from different layers. Furthermore, HMS-Net also improves the sparsity invariant operation in SparseConvs by designing three variants, *i.e.*, sparsity invariant bilinear upsampling, sparsity invariant average, and joint sparsity invariant concatenation and convolution. These new operations are more useful for integrating various levels of feature maps. However, a natural deficiency of using this validity mask is that depth edges would be blurred. In [Jaritz et al., 2018], this problem is attributed to the validity domain extension, *i.e.*, sparse data are "dilated" in each layer with a full convolutional operation [Graham and van der Maaten, 2017]. Consequently, large intensity transitions are averaged and reduced, resulting in blurry edges. This problem always comes along with another issue, *i.e.*, the saturation of the validity mask in early layers. It implies that the validity information cannot be further propagated to subsequent layers. Hence, it may not be necessary to use the validity mask if the network is large enough to extract sufficient features [Jaritz et al., 2018].

Several works have verified the above inference. NASNet [Zoph et al., 2018], a recurrent model initially designed for image recognition, is employed as the encoder

in [Jaritz et al., 2018]. This network is powerful enough in feature extraction such that incorporating the validity mask shows no improvement. Ma et al. [2019] take advantage of residual blocks [He et al., 2016] to extract both low-level and high-level features and fuse them via long skip connections. The two networks have significantly improved depth completion results even without the validity mask but they normally have complicated network structures and involve extensive parameters for training.

In essence, the validity mask is a quite hard constraint because it is binary and the feature activation is biased to valid regions [Jaritz et al., 2018]. This bias makes it harder to differentiate between missing points and zero values, and may degrade the performance in invalid regions to be completed. To soften the constraint, Eldesokey et al. [2019] alternatively propose to treat the validity mask as a continuous confidence field. Using confidence has two primary advantages: (1) Confidence measures the reliability of data, which is more desirable in learning based systems; (2) Confidence can be propagated through the entire network without being saturated in later layers. They calculate confidence from the input by leveraging Normalized Convolution [Knutsson and Westin, 1993]. It also enables the adaptive propagation of confidence throughout the learning process. Output confidence is coherent with the density of input confidence, *i.e.*, pixels within high-density valid regions tend to have higher confidence. The proposed confidence is incorporated into a loss function for the purpose of minimizing prediction errors and maximizing confidence at the same time. A desirable consequence is that reliable points contribute more to final results, further improving the performance with the reduced effect of unreliable depth values.

However, in depth completion, the input is always highly sparse, *e.g.*, valid points only account for 5% in KITTI [Uhrig et al., 2017]. Object structures cannot be appropriately localized due to the sparsity, and the lack of structural cues makes self-guided methods fail to recover semantically consistent boundaries and thin/small objects. This can be improved by using the registered RGB image (captured from the camera) as a reference-guidance, which will be detailed below.

#### 2.2.2.2 Multiple-input (Reference-guided) methods

The RGB image contains dense and rich structural information, *e.g.*, object structures can be easily identified by sharp intensity changes. Existing image guided models take the RGB image as an extra input to the network, and design various approaches to effectively aggregating image features. In the following, we review some representative methods in reference-guided depth completion. They mainly cover three aspects: (1) Plain fusion of image features, *i.e.*, focusing on network architectures that transfer image features to depth features; (2) Enhanced fusion of image features, *i.e.*, making use of some other techniques (not simply modifying networks) to facilitate feature fusion such as confidence propagation, spatial affinity, residual learning, uncertainty, domain adaptation, conditional prior, and the improved loss function; (3) Extra fusion, *i.e.*, leveraging additional cues that are relevant and complementary to

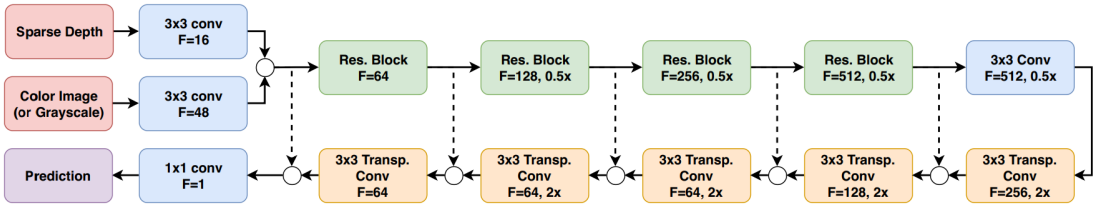



Figure 2.7: Network architecture of supervised Sparse-to-Dense (S2D) [Ma et al., 2019]. This is a standard late fusion framework, where depth and image features are first encoded by two separate networks and then aggregated. Skip connections [He et al., 2016] are used to reduce information loss. Image taken from [Ma et al., 2019].

images like semantic information, 3D point clouds, and surface normal (note that in the literature review, we only care about how existing methods use these cues to improve depth completion rather than the full review of these cues).

**Plain image feature fusion.** In general, there are two types of fusion strategies for image features, *i.e.*, early fusion and late fusion. Early fusion simply concatenates sparse depth and the image in a channel-wise manner, and then feeds them into the network, *i.e.*, they share the same feature encoder [Masci et al., 2013; Ma and Karaman, 2018; Qu et al., 2020]. However, depth and the image are two different modalities, *i.e.*, depth measures distance while the image consists of RGB intensities. It is hard for a single encoder to fit the data and reduce their domain shift [Liu et al., 2020c; Jaritz et al., 2018; Qiu et al., 2019]. To deal with this issue, late fusion is widely used as an alternative, where depth features and image features are extracted with two separate CNNs and then aggregated. The advantage is that the domain shift can be effectively reduced after transforming the two modalities into a similar feature space [Shivakumar et al., 2019]. Jaritz et al. [2018] have justified that late fusion always outperforms early fusion.

A standard late fusion architecture consists of a depth feature encoder, the image feature encoder, and a depth completion decoder (see Fig. 2.7 for a representative late fusion architecture S2D [Ma et al., 2019]). Depth and image features from their corresponding encoders are concatenated or summed as a hybrid feature [Jaritz et al., 2018; Shivakumar et al., 2019]. This is then fed into the decoder for dense depth recovery. Specifically, Jaritz et al. [2018] utilize NASNet [Zoph et al., 2018] to encode richer depth and image features via a larger convolutional kernel size. Shivakumar et al. [2019] alternatively leverage Spatial Pyramid Pooling [He et al., 2015] to learn coarse-to-fine feature representations by taking both local and global information into account. They also empirically find that the depth encoder should have larger kernel sizes with fewer convolutional layers to cover most missing regions. By contrast, the image encoder should have smaller kernel sizes with more convolutional layers to better extract low-level image features, *e.g.*, edges. However, as Liu et al. [2020c] point out, late fusion may suffer from information loss during feature propagation (features are gradually compressed) and limited feature interaction (features from two modalities interact only at the end of the encoder). Both factors are likely to lead

to sub-optimal performance.

To address the above issue, skip connections [He et al., 2016] are embedded into the network to reduce information loss [Ma et al., 2019; Fu et al., 2020; Schuster et al., 2021]. Yan et al. [2020] use skip connections to link the encoder and decoder, *i.e.*, transferring low-level and mid-level depth and image features to the decoder. To boost feature interaction, it is also useful to build connections between intermediate features of depth and image encoders. Huang et al. [2019] first employ six blocks of ERFNet [Romera et al., 2017] that consists of two downsampling blocks and four residual blocks to extract image features, and then concatenate them into depth features at multiple scales.

Although feature concatenation is simple to implement, it fuses the two types of features as a whole and cannot provide discriminative image cues that specially complement depth features. To learn more selective image features, Tang et al. [2020] take advantage of the guided image filter [He et al., 2013] and design a guided convolution module to generate content-dependent (kernel weights are dynamically varied based on image content) and spatially-variant (kernels also change with different spatial positions) kernels. This operation enables the model to adaptively acquire more specific image features. Lee et al. [2020] propose the concept of "cross-guidance", which means depth and image features can guide and complement each other. This is achieved by introducing a learnable attention map and refining features with attention weights. Depth and image features, in that case, can retain their own distinctive features, and simultaneously improve the feature representation by incorporating complementary information from the other. A multi-scale guided framework is introduced in [Li et al., 2020a], where sparse depth is first downsampled into lower resolutions, *i.e.*, 1/2 and 1/4, and the two resized maps together with the original input are fed into three cascade Hourglass networks [Newell et al., 2016]. Multi-scale image features, extracted by a separate encoder, are added to corresponding depth features with identical scales. The distinction is that predictions from three Hourglass networks are all supervised by dense depth ground truth. The final output is improved with the integration of features at multiple scales.

Hu et al. [2021] study depth and image features from a new perspective, *i.e.*, extracting and fusing depth-dominant and image-dominant features by modifying the network input. Specifically, two branches are designed, *i.e.*, color dominant and depth dominant, and they have the same encoder-decoder architecture. The color dominant branch takes the image and sparse depth as input, and outputs a raw dense depth map. This raw map is combined with sparse depth and then fed into the depth dominant branch for another dense prediction. Moreover, the decoder features of the color branch are concatenated with the depth encoder features at various scales. The two branches also output corresponding weighting maps to adaptively merge two dense outputs. This model comprehensively covers feature fusion, feature selection, and output refinement.

In summary, plain image feature fusion methods focus on the design of network architectures to fuse depth and image features. It has been demonstrated that incorporating image features can significantly improve depth completion performance.

Selective/Adaptive feature fusion is currently a hot topic in this filed.

**Confidence propagation.** As introduced in Section 2.2.2.1, confidence measures the reliability of output depth and can be learned and propagated through the network training. The necessity of using confidence owes to the fact that input LiDAR values are always noisy. Noise is accumulated after projecting raw LiDAR point clouds to the image plane, *e.g.*, mixed depth values around occluded object boundaries [Gu et al., 2021; Qiu et al., 2019]. Since the image can provide complementary structural cues to depth, confidence measurement becomes more faithful after being combined with image information. The usage of confidence is generally divided into two types, *i.e.*, output fusion and intermediate constraint (detailed below).

Output fusion means the confidence map works as a per-pixel weighting function to fuse various outputs for the final prediction (the sum of weights for all pixels is equal to 1). To generate different outputs, Van Gansbeke et al. [2019] design a global branch that takes the image and sparse depth as input and a local branch that only inputs sparse depth. The global branch highlights image features to reduce incorrect predictions caused by input depth noise and enforces depth smoothness. The local branch focuses more on depth features, which are further refined by global branch features. In essence, the global branch functions as a prior to regularize the local one. Both branches output a dense depth map, and the two maps are fused as per the predicted confidence. A similar strategy is employed in [Lee et al., 2020]. Hu et al. [2021] merge outputs from the color-dominant branch and the depth-dominant branch. A general finding from aforementioned works is that depth confidence presents higher values in most regions while image confidence is higher around object boundaries. Note that the confidence map is learned by the network itself.

Intermediate constraint refers to the confidence map is applied in the intermediate step to constrain features, *i.e.*, only features with high confidence can be further propagated. Intermediate confidence can be either automatically generated and updated within the network or trained with ground truth confidence. For automatically-learned confidence, Park et al. [2020] combine the image and sparse depth as a hybrid input to an encoder-decoder network. The predicted confidence map from that network further facilitates the learned affinity (depicting correlations between pixels). Eldesokey et al. [2019] first use a multi-scale unguided network to produce the confidence map, and then concatenate it with the image as the input to the feature extraction network. The input to the unguided network for confidence estimation is a binary mask, which may cause undesirable artifacts in recovered depth [Eldesokey et al., 2020]. This can be improved by learning input confidence in a self-supervised manner [Eldesokey et al., 2020]. Qiu et al. [2019] utilize two types of confidence, one for intermediate constraint and the other for output fusion. For the trained confidence, Xu et al. [2019] generate confidence ground truth by calculating the Laplace difference between input values and depth ground truth. An additional loss that penalizes confidence is added as part of the training loss. Xu et al. [2020] also train confidence in this way.

In summary, using confidence can largely lower the negative impact of noisy values in input depth. However, this method, in essence, prevents low-confident points

from propagating through the network, which may further increase the sparsity of the depth map, *i.e.*, reducing the number of available depth points. A potential solution is to correct these noisy values directly in a data-driven manner prior to network training.

**Spatial affinity.** Spatial affinity generally reflects dense and global pairwise relationships of an image, *i.e.*, the similarity between two pixels [Liu et al., 2017a]. In depth completion, the recovery of depth largely relies on neighbouring points. The lack of this contextual information would result in blurry edges or structure misalignment [Cheng et al., 2018]. Hence, it is important to figure out the appropriate spatial affinity for propagating context.

Spatial Propagation Network (SPN) [Liu et al., 2017a] adopts three-way connection in four directions, *i.e.*, left-to-right, top-to-bottom, and their opposite directions. The spatial affinity is learned through a network in a data-driven manner, which is more effective than hand-crafted affinities. However, SPN cannot involve all the neighbouring information at the same time, so it is less robust to the sparsity and irregular distribution of the input [Park et al., 2020]. To deal with this issue, Convolutional Spatial Propagation Network (CSPN) [Cheng et al., 2018] updates all pixels within a local field simultaneously. It is also able to capture the long-range dependency via a recurrent architecture. Nevertheless, CSPN treats all the local pixels equally, which cannot give more specific attention to some key regions such as boundaries. Moreover, the recurrent operation occupies substantial computational resources. CSPN++ [Cheng et al., 2020] improves CSPN by introducing context and resource awareness. Context awareness is achieved by assembling the outputs from multiple convolutional kernel sizes and different iterations. Based on this, the network sequentially selects a kernel size and a number of iterations for each pixel, which is adaptive to various image content and significantly reduces computational costs. One common issue of SPN, CSPN, and CSPN++ is that their local field for calculating the affinity is fixed, in which case some farther but useful context cannot be well captured and some irrelevant information within the local region may be inevitably retained.

To enlarge the receptive field and reduce the impact of irrelevant local neighbours, Non-Local Spatial Propagation Network (NLSPN) [Park et al., 2020] is proposed for the purpose of (1) predicting non-local neighbours for each depth point, and (2) integrating relevant features with spatially-varying affinities. Deformable Spatial Propagation Network (DSPN) [Xu et al., 2020] uses an offset estimator to generate the offset for each pixel so that the receptive field for affinity propagation is increased and becomes adaptive. Adaptive Context-Aware Multi-Modal Network (ACMNet) [Zhao et al., 2021] makes use of the graph network [Wang et al., 2019b] and obtains adaptive contextual information via graph propagation.

In summary, spatial affinity improves depth completion by taking more contextual information into account. The fundamental problem is how to make the receptive field for affinity propagation more adaptive to image content, especially around object boundaries. Recent progress on non-local and deformable propagation can inspire more research into this direction.

**Residual learning.** Most methods treat depth completion as a one-stage task, *i.e.*, mapping directly from the sparse input to the dense output. This framework may have two limitations: (1) Depth pixels without input values are filled with zero, which increases the ambiguity between valid pixels that have depth values and these zero-filled pixels [Liu et al., 2021; Dimitrievski et al., 2018; Liao et al., 2017]; (2) The network has to first eliminate the aforementioned ambiguity, which limits its capacity in learning sufficient features.

To solve above issues, the residual learning framework is employed, which is derived from the residual network [He et al., 2016]. It basically has two stages: (1) Sparse-to-coarse, *i.e.*, transforming the sparse input to a coarse dense depth map; (2) Coarse-to-fine, *i.e.*, refining the coarse map by adding the learned depth residual to it [Liu et al., 2020b]. In essence, residual learning recovers high-frequency depth such as boundaries and some details, which are not always accurately completed in one-stage methods [Gu et al., 2021]. In the sparse-to-coarse stage, coarse depth is computed from the sparse input via the nearest neighbour interpolation [Chen et al., 2018], morphological operations [Gu et al., 2021], kernel regression [Liu et al., 2021], or some simple depth completion approaches [Liu et al., 2020b]. The coarse-to-fine stage normally takes advantage of more powerful networks or techniques to learn the depth residual more effectively, *e.g.*, DenseNet blocks [Huang et al., 2017] used in [Chen et al., 2018], Depth Completion Unit (DCU) [Qiu et al., 2019] used in [Gu et al., 2021], UNet [Ronneberger et al., 2015] used in [Liu et al., 2021], channel shuffle [Zhang et al., 2018] used in [Liu et al., 2020b]. Note that coarse dense depth can also be replaced by other dense representations, *e.g.*, the plane-residual (representing the distance from the closest pre-defined discretized depth planes) [Lee et al., 2021], reference depth (a group of lines constituting the surface vertical to the ground) [Liao et al., 2017].

In summary, residual learning enables depth completion from coarse to fine, which reduces the impact of zero-filled pixels. It also effectively refines and improves coarse depth, especially around boundaries.

**Uncertainty.** Although CNNs have been successfully applied to various vision tasks, there is little interpretation on how the network makes predictions and how reliable these predictions are. The uncertainty is quantified by leveraging the Bayesian Neural Network to output the probabilistic distribution parameterized by mean and variance [Kendall and Gal, 2017]. The uncertainty in depth completion mainly comes from the noisy nature of the sparse input [Eldesokey et al., 2020] but few works investigate the uncertainty measure. Eldesokey et al. [2020] fill in this gap by introducing uncertainty analysis to depth completion. By modelling the variance of input noise and combining it with output confidence, they quantify the uncertainty of predictions, *i.e.*, output confidence is strongly correlated with prediction errors. This is a useful method to measure the reliability of output depth values.

Note that although the uncertainty is also mentioned in [Van Gansbeke et al., 2019], it refers to the difference between the predictions from depth-only and RGB-D branches, which is not the probabilistic analysis as in [Eldesokey et al., 2020].

**Domain adaptation.** Considering dense depth ground truth is hard and expen-

sive to acquire in practice, Lopez-Rodriguez et al. [2020] train the model on synthetic data and adapt it to real-world cases. To reduce the LiDAR gap, CARLA [Dosovitskiy et al., 2017], an autonomous driving simulator, is employed to generate synthetic LiDAR that has similar input distribution as the real data. Moreover, it simulates the see-through artifacts by setting up a multicamera environment and projecting the virtual LiDAR to the RGB image frame. For RGB, they make use of CycleGAN [Zhu et al., 2017] for style transfer from the real domain to synthetic examples.

**Conditional prior.** It is a fact that the correlation between the RGB image and dense depth is stronger than that between the image and sparse depth, because the image itself is dense. Yang et al. [2019] argue that the performance of directly combining the image with sparse depth would be less good than exploiting a prior from the image and its corresponding dense depth. They make use of the Conditional Prior Network (CPN) [Yang and Soatto, 2018], and calculate the prior by taking the image and dense depth to the network on the Virtual KITTI dataset [Gaidon et al., 2016]. This prior represents the conditional probability of dense depth given the image, and generates the posterior estimate of depth after being incorporated into a likelihood term. The proposed approach is beneficial for analyzing the underlying probabilistic relationship between the image and depth, but requires additional resources to train the CPN in advance.

**Improved loss function.** The depth mixing problem, *i.e.*, ambiguous depth values between the foreground and background, is a fundamental challenge in depth completion [Imran et al., 2019, 2021; Qiu et al., 2019]. It leads to discontinuous boundaries or distorted object shapes in the output. Popular loss functions for training the completion model, *e.g.*, MSE and MAE, would even promote depth mixing [Imran et al., 2019]. To solve this problem, Imran et al. [2019] propose a novel representation for depth, *i.e.*, Depth Coefficients (DC). The general idea is to convert sparse depth into multiple channels, and each channel represents a fixed depth range. Any depth value is the weighted sum of these channel bins. The discrete nature of DC largely preserves the depth continuity, so that depth mixing can be reduced. Another advantage of DC is that it represents depth in a probabilistic way, and the model can be trained with the cross-entropy loss. Using this loss can accelerate the convergence speed, and more importantly, reduce depth mixing. However, converted channels share the identical resolution with the input, and always have large computational costs and occupy substantial memory [Imran et al., 2021]. Instead of relying on multiple channels, Imran et al. [2021] alternatively introduce a two-surface representation, *i.e.*, foreground depth and background depth. This is more intuitive and efficient. Moreover, a pair of asymmetric loss functions [Vogels et al., 2018] are employed to focus on foreground and background depth respectively, which is beneficial for separating ambiguous depth values. A fusion module is incorporated for final prediction, *i.e.*, adaptively selecting foreground or background depth in ambiguous regions and fusing the two depth in other regions. Both methods exploit new depth representations to handle the depth mixing problem, and design corresponding loss functions to further improve the accuracy of depth completion.

**Use of semantic information.** Although RGB images can provide structural

cues that facilitate depth completion, they may suffer from the sensitivity to optical changes and complex textures. For example, a car may present different colors due to reflection or shadows. Differently, semantic labels are more robust to these factors. Motivated by this, Zhang et al. [2021] combine semantic segmentation and depth completion as a multi-task learning framework. The two tasks are trained with separate networks and synergized via the semantic guided smoothness loss. This method enhances the consistency between semantic and geometric structures, which effectively improves the robustness to various scenes.

**Use of 3D point clouds.** Depth is highly associated with the scene geometry, but the RGB image cannot explicitly provide geometric cues [Fu et al., 2020; Wong et al., 2021a]. 3D geometric information gives better discrimination of occlusion boundaries and foreground/background objects, which is complementary to RGB and useful for depth completion [Chen et al., 2019]. Initial LiDAR scans are 3D point clouds, but they cannot be directly convolved with standard CNNs. Hu et al. [2021] design a geometric convolutional layer that concatenates a 3D position map (three maps including X, Y, and Z) to the input of the layer. Chen et al. [2019] employ continuous convolutions [Wang et al., 2018] performed on 3D points. Afterwards, the learned 3D features are fused with 2D appearance features through the 2D-3D fuse block. They demonstrate that leveraging geometric cues enables the model to recover sharper and more accurate object boundaries.

**Use of surface normal.** Another widely-used geometric cue is the surface normal, which is determined by the tangent plane of the local surface [Chen and Schmitt, 1992]. The surface normal can be estimated from depth, and depth can also be inferred from the surface normal by linear operations [Qi et al., 2018]. Hence, the surface normal and depth are strongly interrelated [Xu et al., 2019], and using the normal to guide depth can reduce depth distortion in planar regions [Qi et al., 2018]. Zhang and Funkhouser [2018] first predict the surface normal and occlusion boundaries from the RGB image. Considering the normal and boundaries are local properties of the surface, they are then combined with sparse depth, and the dense output is produced by optimizing a globally defined linear function. In essence, this method is a post-processing step for the sparse input. However, the surface normal is separately generated, and depth-normal correlations are not well exploited [Xu et al., 2019]. To enhance their correlations, Qiu et al. [2019] design two pathways, *i.e.*, the color pathway and the surface normal pathway, and produce two dense depth maps from them separately. The final output is the weighted sum of the two maps, and the weights are automatically learned by the network. The two pathways are jointly trained so that surface normal estimation and depth completion can facilitate each other. Xu et al. [2019] study depth-normal constraints by modelling their locally linear orthogonality in the plane-origin distance space (the distance from the corresponding tangent plane to the camera centre). To maintain the consistency between depth and the surface normal, Lee et al. [2019a] introduce the depth-normal consistency loss to minimize their inner product. The effectiveness of the surface normal has been well justified in depth completion. How to better aggregate depth and surface normal features is worthy of further exploration.

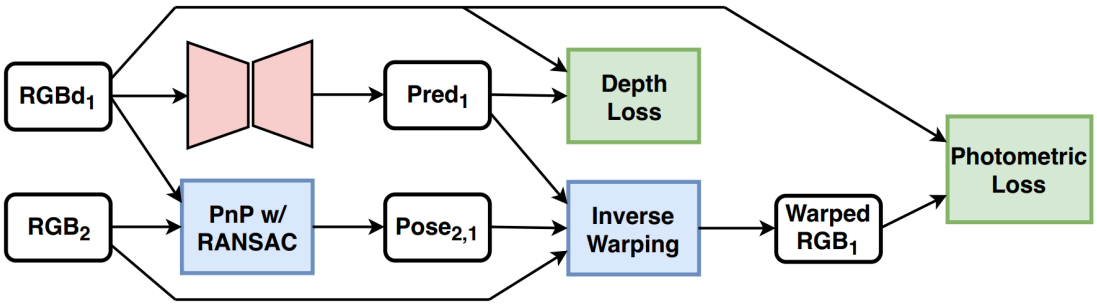



Figure 2.8: Network architecture of unsupervised Sparse-to-Dense (S2D) [Ma et al., 2019]. In this framework, the registered image is combined with sparse depth. During training, sparse depth is used as a supervision signal, and a second image is required to construct the photometric loss. Image taken from [Ma et al., 2019].

### 2.2.3 Unsupervised Learning Methods

Unsupervised depth completion aims to recover dense depth from the sparse input without the supervision of dense depth ground truth[1]. The pioneering work [Ma et al., 2019], shown in Fig. 2.8, provides four basic components for the unsupervised learning framework:

(1) **RGB image as input.** The RGB image is taken as an additional input to the network.

(2) **Sparse depth supervision.** The sparse input itself is used as a supervision signal for depth by penalizing the difference between the input and output in valid pixels (with depth values in the input).

(3) **Photometric loss.** Sparse depth cannot supply per-pixel supervision due to its high sparsity [Zhang et al., 2019a]. To deal with this issue, stereo information, *e.g.*, a left-right image pair [Yang et al., 2019] or temporally consecutive video frames [Ma et al., 2019; Wong et al., 2020, 2021a,b], can implicitly give dense supervision to depth. Specifically, the left image (or Frame 1) can be transformed into the right image (or Frame 2) based on the depth from the right view (or Frame 2), and vice versa. The transformation process is different under the two settings. The use of the left-right image pair generally assumes a displacement (also known as *disparity*) exists between the two images, which is formulated as an intensity constancy constraint. Depth can be computed from the disparity based on the focal length and the baseline of left and right cameras. Consequently, we can approximate the second image from the first one with the predicted disparity (or depth). Differently, using video frames for constructing the photometric loss requires pose estimation first, *i.e.*, calculating the relative pose between two frames. Given the intrinsic matrix of the camera, the second image can be warped to the first one according to the relative transformation and estimated depth.

[1]Although non-learning methods in section 2.2.1 do not use ground truth either, they are not in the same category of unsupervised learning methods because they do not involve any network training. In this thesis, "unsupervised depth completion" refers to learning-based depth completion without ground truth.

Hence, the quality of the transformed image is largely dependent on the depth map. This, in turn, indirectly supervises depth completion by penalizing the photometric loss between the transformed image and the target one. This supervision is implicit because it does not directly penalize depth, but the result derived from it, *i.e.*, the transformed image. The second image is not required during test time.

(4) **Smoothness loss.** This loss aims to reduce local discontinuities and sharpen boundaries in output depth.

Subsequent works [Wong et al., 2020; Yang et al., 2019; Wong et al., 2021a,b] follow this framework but have their own distinctions. Wong et al. [2020] add SSIM [Wang et al., 2004] to the photometric loss to enhance the robustness to local illumination changes and the sharpness of boundaries in color images, similar to [Yang et al., 2019] and [Wong et al., 2021a]. Yang et al. [2019] make use of the dense depth prior obtained from CPN [Yang and Soatto, 2018] to model the likelihood and conditional distribution of depth. Furthermore, they leverage the disparity loss combined with left-right image transformation. Wong et al. [2021a] learn the scene topology as a prior from synthetic data, which has good generalization to various scene geometry. It is also incorporated as part of the loss function, aiming to enhance the compatibility of the prior and predicted depth with the image. Wong et al. [2021b] treat sparse depth supervision as the data fidelity term, and other losses as the regularization term. They observe that in occluded regions, the disparity is not correct, so the depth penalty there is uninformative and should be discarded. This can be compensated by increasing the impact of regularization. Another observation is that depth errors in homogeneous regions with large disparities are easy to minimize, in which case the regularization naturally has more impact on completion results. Motivated by the two cases, they introduce an adaptive weighting scheme that varies over the image domain and training epochs. It consists of (1) the weight for the data fidelity term determined by the probability of a given pixel co-visible in two images; and (2) the weight for the regularization term determined by the depth residual at each pixel over each training step. In essence, the two weights are complementary to each other, and they can be conveniently incorporated into existing unsupervised models and improve their performance.

In summary, unsupervised depth completion has achieved good progress since 2019. However, existing methods heavily rely on RGB images for additional input and constructing the photometric loss. In this thesis, we explore how to realize unsupervised depth completion only from the sparse input.

## 2.3 Summary

In this chapter, we have reviewed existing literature on image smoothing and depth completion respectively. For image smoothing, we have introduced kernel methods, global methods, and deep learning methods. In each type, we have further studied the usage of self-guidance and reference-guidance. For depth completion, we have given an overview of non-learning methods, supervised learning methods, and

unsupervised learning methods. We have also investigated self-guided (depth-only) and reference-guided (multiple-input) settings. Through the extensive literature review, we can have a comprehensive understanding of the two tasks, including their history, mainstream methods, and up-to-date progress.

Chapter 3

# Kernel-Based Double-Guided Filter

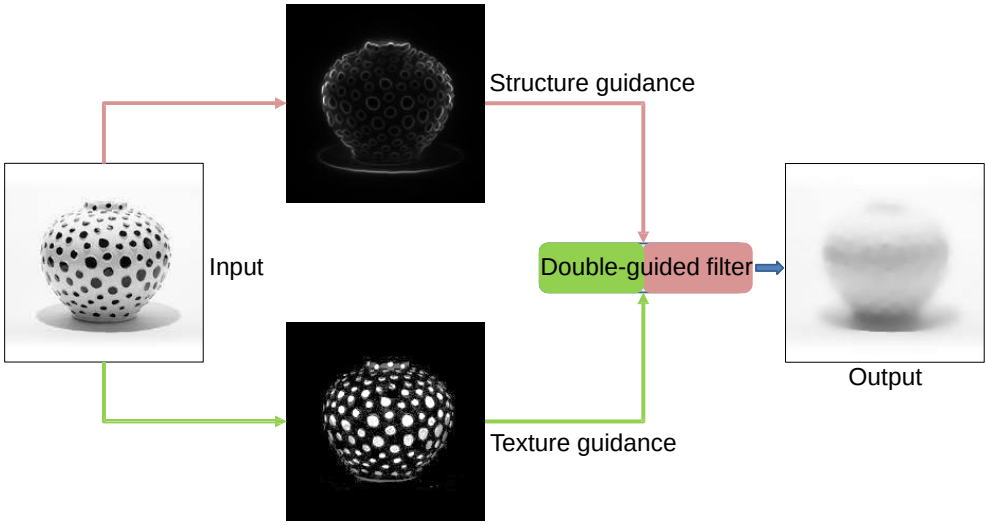
From the following two chapters, we will introduce our research work on image smoothing, *i.e.*, retaining main structures and removing insignificant textures in the image. Existing methods make use of structure guidance by generally assuming structures have large intensity difference or gradients. However, this assumption is not reliable when there exist strong textures with large color contrast. In that case, some structures may be blurred as the filter attempts to remove these textures. Also, some strong textures are mistakenly retained as structures. In our work, we aim to improve the discrimination of structures and textures by using independent structure guidance and texture guidance so as to reduce structure degradation/edge blurriness. We develop both the hand-crafted kernel filter (this chapter) and deep filter (Chapter 4).

In this chapter, we propose a novel kernel-based double-guided filter (DGF). To enhance the discrimination of textures, for the first time, we introduce the concept of "texture guidance" to indicate the position and magnitude of textures. Additionally, we adopt semantic edge detection as structure guidance, which is beneficial for preserving more semantically meaningful structures. The proposed DGF incorporates the two forms of guidance into the kernel operation to be both "structure-aware" and "texture-aware". Through extensive experiments, we provide the appropriate usage of the DGF and demonstrate that it can effectively remove strong textures without blurring main structures.

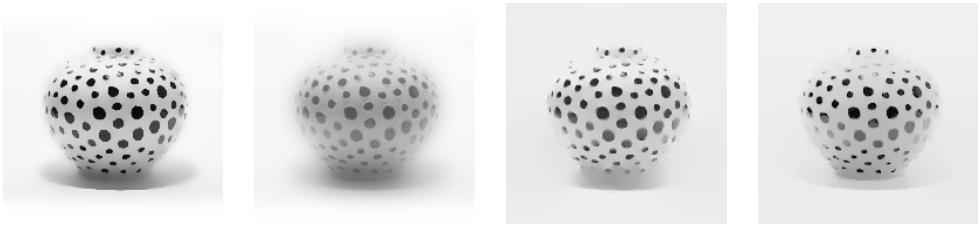
In the remainder of this chapter, Section 3.1 introduces our motivation of using structure guidance and texture guidance, and summarizes primary contributions. Section 3.2 further illustrates our motivation and the generation of structure guidance and texture guidance. We provide details of the proposed DGF in Section 3.3. Section 3.4 discusses the parameter setting of the DGF, and compares our filter with other methods in terms of visual results, denoising performance, and three typical applications. Section 3.5 summarizes the chapter and proposes the future work.

## 3.1 Introduction

Image smoothing, which aims to preserve main structures/edges and remove insignificant details/textures, plays an important role in many computer vision ap-




(a) Framework of proposed double-guided filter



(b) BLF (c) GF (d) L0 (e) WLS

Figure 3.1: Framework of the proposed double-guided filter (DGF) and smoothing results with different methods. (a) DGF utilizes two independent structure and texture guidance for better discrimination of structures and textures. (b)-(e) Dotted textures on the vase are strong and they are mistakenly retained as structures by existing methods, *e.g.*, BLF [Tomasi and Manduchi, 1998], GF [He et al., 2013], L0 [Xu et al., 2011], WLS [Farbman et al., 2008]. Also, the main structures, especially the base of the vase, are severely blurred in GF, L0 and WLS. Our DGF can remove these strong textures and preserve main structures at the same time.

plications, such as image abstraction [Winnemöller et al., 2006], detail enhancement [Fattal et al., 2007], image denoising [Gu et al., 2014].

Existing image smoothing methods can be roughly classified into two types: kernel-based local filtering, and global-based structure and texture separation. Both types of methods largely focus on "structure-awareness". For example, the bilateral filter (BLF) [Tomasi and Manduchi, 1998] and guided filter (GF) [He et al., 2013] calculate a local average of intensities by convolving with a positive kernel. This operation can retain large gradients by adjusting weights of neighboring pixels according to their intensity difference. The averaging operation is able to suppress weak textures (small oscillations in intensities) effectively. However, as Zhang et al. [2015] point out, the essential deficiency of this type of method is the lack of discrimination of strong textures (insignificant details with high contrast) and structures. For

example, as shown in Fig. 3.1, the input image contains a vase covered with black dots. The removal of these black dots will not affect our recognition of the vase, thus we regard them as insignificant details that can be removed. However, since their contrast is very high, they are mistakenly regarded as structures (see filtering results of the BLF and GF in Fig. 3.1(b) and (c)). Global-based separation methods extract textures from the input image by optimizing a globally-defined objective function. This is based on the assumption that an image can be decomposed into a structure layer and a texture layer, and structures tend to have larger gradients. Hence, penalizing gradients is a normal setting to regulate the smoothing process. For example, L0-smoothing [Xu et al., 2011] manipulates the total number of non-zero gradients, and WLS [Farbman et al., 2008] leverages the total variation between two layers in terms of gradients. As Fig. 3.1(d) and (e) show, the overall structure of the vase is over-smoothed when the two methods attempt to remove dotted textures (the over-smoothing is especially severe at the base of the vase, which has low contrast but important semantic meaning). Therefore, only relying on intensity difference or gradients is not always reliable in differentiating between structures and textures.

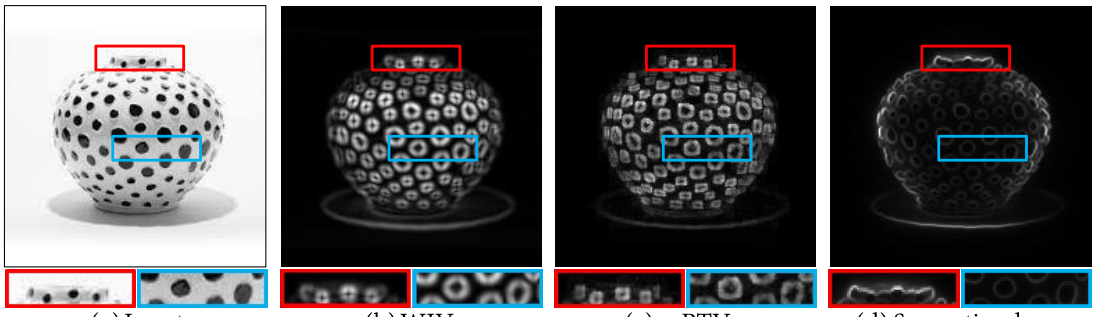
Our idea is to leverage two forms of independent guidance, *i.e.*, structure guidance and texture guidance, to infer structures and textures respectively. To this end, we design a double-guided kernel-based filter (**DGF**). It is able to preserve meaningful structures with the guidance of the newly-proposed semantic edge detection method [Hallman and Fowlkes, 2015] (**structure guidance**), and distinguish and remove textures with the guidance of image separation [Liu et al., 2013b] (**texture guidance**). Fig. 3.1(a) illustrates the framework of the DGF. It achieves better smoothing results than other methods in the vase example. More importantly, the proposed DGF is easy to use because the kernel only involves two parameters that correspond to "structure-awareness" and "texture-awareness".

In summary, we make the following major contributions:

- We give theoretical insights into balancing "structure-awareness" and "texture-awareness" for image smoothing.
- It is the first time that structure guidance and texture guidance are applied to image smoothing at the same time. Furthermore, the two guidance maps are generated independently.
- The proposed easy-to-use double-guided filter outperforms existing methods by simultaneously achieving "structure-awareness" and "texture-awareness". Hence, it can remove even stronger textures without blurring main structures.

## 3.2 Structure Guidance and Texture Guidance

To the best of our knowledge, most existing image smoothing methods only depend on structure guidance. However, this is not sufficient because in many case textures may also have strong edges, which will confuse the structure guidance map. Thus, we need texture guidance to tell the filter where to remove (smooth more), especially



(a) Input (b) WIV (c) mRTV (d) Semantic edges

Figure 3.2: Structure confidence maps of the "*Vase*" example. From left to right: input, structure map calculated from [Xu et al., 2012], structure map calculated from [Cho et al., 2014], semantic edge map [Hallman and Fowlkes, 2015]. The semantic edge detection can help to form meaningful edges that are closer to human perception. It also outperforms other algorithms that simply use gradients to differentiate between structures and textures.

when the filter encounters strong textures. We note that separation-based methods decompose the image into structure and texture layers, which achieve a better trade-off between preserving structures and removing textures than kernel filters. Here we use the texture layer to guide local filtering. In essence, texture guidance reduces the possibility of retaining strong textures. The two forms of guidance will be introduced in the following.

### 3.2.1 Structure Guidance

Ideal structure guidance should give high confidence to meaningful structures, no matter their gradients are large or small. Moreover, it is expected to give relatively low confidence to insignificant textures regardless of their gradients either. As we know, humans can easily distinguish textures from structures due to the advanced cognition system built in our brains. Semantic edge detection makes structure perception closer to humans because it is based on learning from a large number of human-labelled images. We apply the recently proposed approach [Hallman and Fowlkes, 2015] to generate semantic structure guidance. The confidence map $E$ is obtained by normalizing the detection result into 0-1, as shown in Fig. 3.2(d).

It should be noted that the choice of structure guidance is an open question. To demonstrate the effectiveness of our choice of semantic guidance, we compare it with other two gradient-based approaches: Windowed Inherent Variation (WIV) [Xu et al., 2012] and modified Relative Total Variation (mRTV) [Cho et al., 2014]. Specifically, WIV constructs the confidence map by considering the gradients of all the neighboring pixels, and mRTV extends it by taking intensity difference into account. We refer the reader to the two papers for more details.

Edge confidence maps from three approaches are shown in Fig. 3.2 and we select representative close-ups for further illustration. In the "*Vase*" example, with semantic edge detection, there forms a clear boundary at the top of the vase, which cannot be

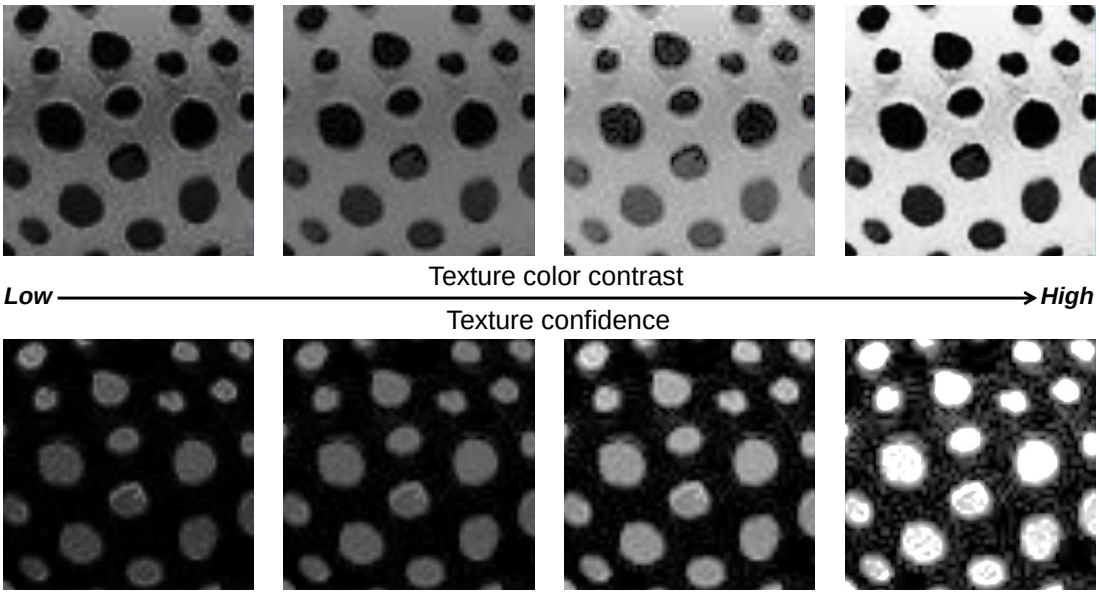



Figure 3.3: Illustration of texture guidance. The texture confidence map indicates both the position and magnitude of textures. Larger magnitude (or color contrast) of textures corresponds to higher confidence, which means the textures are stronger and harder to remove.

achieved by the other two methods. Also, for textures on the surface, although the semantic detection result cannot fully remove the negative influence of extremely high contrast, it effectively weakens texture edges. In essence, MIV and mRTV are both derived from gradients directly, which limits their ability in differentiating between structures and textures.

### 3.2.2 Texture Guidance

Ideal texture guidance should indicate both the position and magnitude of textures. This can be revealed by a texture confidence map. As shown in Fig. 3.3, the position of textures can be easily observed from the confidence map as non-textured regions tend to have zero confidence. The magnitude of textures is reflected by the specific confidence value, *i.e.*, larger confidence corresponds to stronger textures that are harder to remove. It is important to know that texture guidance does not directly reflect structure information as structure guidance does, it provides special estimation of textures. This, in turn, improves structure preservation by reducing the confusion between structures and textures.

To generate texture guidance, we take advantage of global methods as they show better robustness in extracting textures than kernel approaches. We note that although the SGTD algorithm [Liu et al., 2013b] is also based on the gradient assumption (see Section 3.1), it explores a new way to minimize the correlation between structure gradients and texture components (magnitude). The two measurements are different in nature so that we can generally think the two layers are independently

generated. This method has shown superior performance over existing separation-based methods in removing textures, which can be used to construct texture guidance.

In detail, given the input image $I$, the output $S^*$ is generated as follows:

$$S^* = \arg\min_S \sum_p (S_p - I_p)^2 + \left|\nabla S_p\right| + \left|S_p - I_p\right| \cdot \left|\nabla S_p\right|, \tag{3.1}$$

where $\nabla S_p = \sqrt{(\partial_x S_p)^2 + (\partial_y S_p)^2}$ is the gradient at pixel $p$. We calculate the magnitude of the texture layer $|S^* - I|$ and normalize it to 0-1, and then get the texture confidence map $T$.

## 3.3 Double-Guided Filter

Based on the analysis above, we define the double-guided filter (DGF) which only relies on one parameter for each guidance. Given the input $I$[1], the DGF is defined as:

$$\tilde{I}_p = \frac{1}{\kappa_p} \sum_{q \in \Omega} w_q^T w_{pq}^E \cdot I_q, \tag{3.2}$$

where $\Omega$ is a $k \times k$ squared kernel centered at $p$. $\kappa_p$ is used for normalization. $w_{pq}^E$ and $w_{pq}^T$ represent the structure weight and texture weight respectively, which are detailed below.

### 3.3.1 Structure Weight

Given structure guidance $E$, $w_{pq}^E$ takes the form of:

$$w_{pq}^E = (1 - E(q)) \cdot \exp\left(\frac{-\|I(p) - I(q)\|^2}{2\sigma_s^2}\right), \tag{3.3}$$

where $\sigma_s$ is a user-specified parameter. The right part of the structure weight is the range kernel found in the bilateral filter [Tomasi and Manduchi, 1998], which modulates smoothing by intensity difference. This kernel essentially retains strong textures in the bilateral filter because both main structures and strong textures have large intensity difference. To improve it, we multiply it by $(1 - E(q))$, so that *intensity difference will only be retained if the corresponding edge confidence is high*. Even though some part of the structures is weak but with semantic meaning, this guidance can give more confidence and lower the weight to preserve intensity difference.

[1]The input can be either a gray or a RGB image. If it is an RGB image, the filter is performed on each color channel separately.

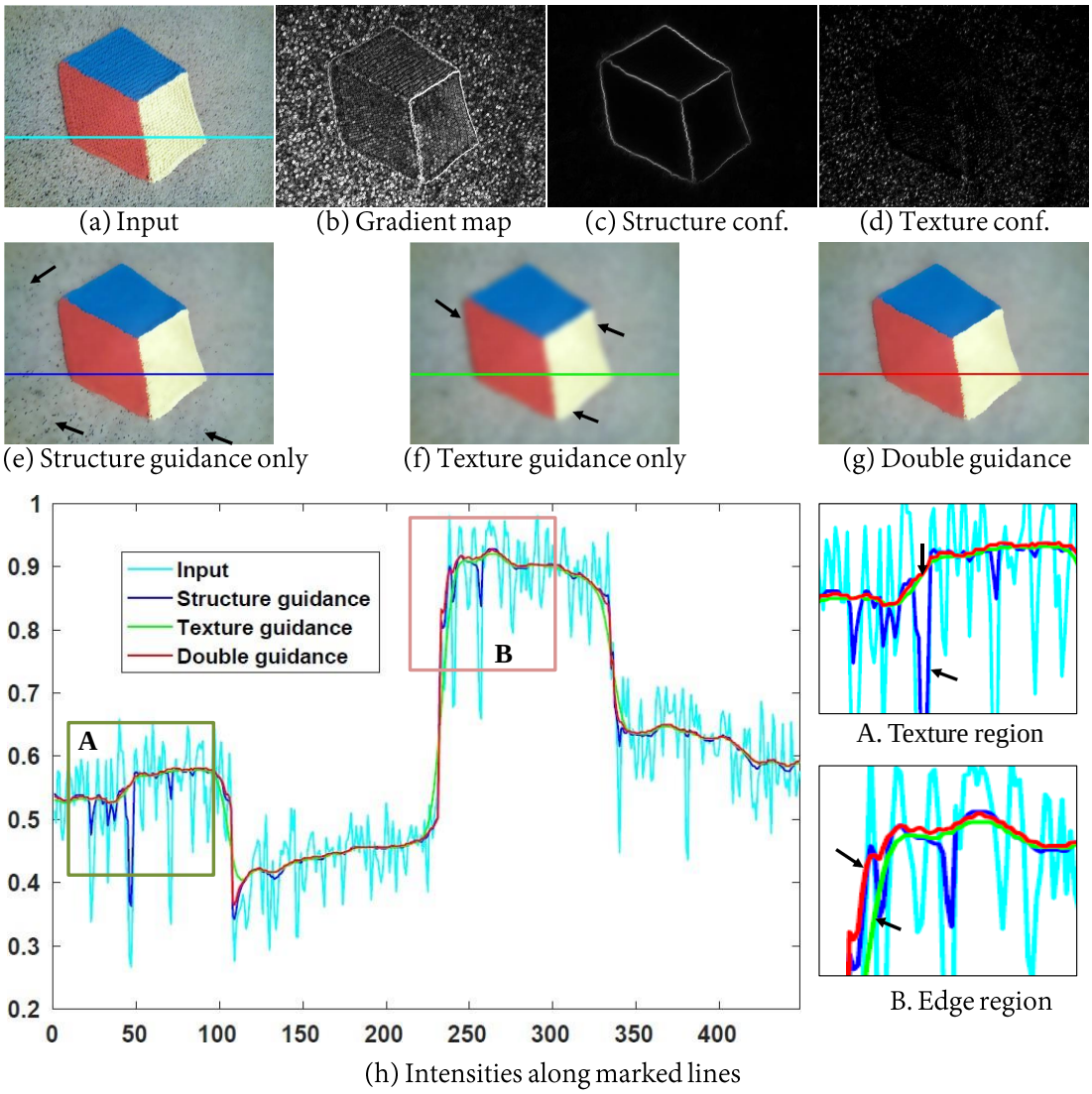



Figure 3.4: Illustration of the double guidance process. The gradient map widely used by existing methods is largely affected by textures. The semantic structure map we use can reflect more semantically meaningful structures. Only using structure guidance cannot fully get rid of the influence of strong textures and only using texture guidance will blur main structures. The combination of two guidance yields a better smoothing result in both structure preservation and texture removal.

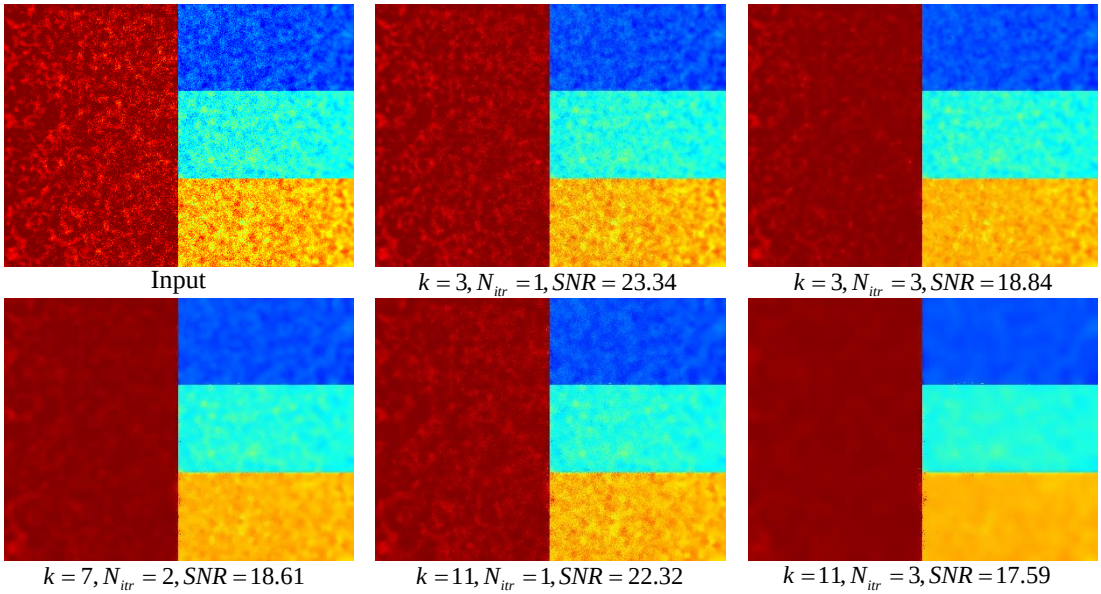



Figure 3.5: Double-guided filtering with different kernel sizes and iterations. A larger kernel size and more iterations make it easier to suppress textures.

### 3.3.2 Texture Weight

Given texture guidance $T$, $w_q^T$ is defined as:

$$w_q^T = \exp\left(\frac{-T(q)^2}{2\sigma_t^2}\right), \tag{3.4}$$

where $\sigma_t$ is a user-specified parameter. Intuitively, pixels with high texture confidence should be smoothed without affecting structure pixels, so they are assigned with small weights. Other non-texture pixels have relatively larger weights to facilitate smoothing out textures.

### 3.3.3 Effect of Single and Double Guidance

The highlight of the proposed DGF is that structure guidance and texture guidance support each other in preserving structures and removing textures. To illustrate this, Fig. 3.4 shows an example and its smoothing results with single structure or texture guidance and double guidance. Intuitively, the result only using structure guidance preserves edges but some strong textures are still retained. By contrast, the result only using texture guidance does not contain textures but overall structures appear blurred. To visualize the effect, we plot the color intensity distribution along one line (marked in the image) in Fig. 3.4(h). We first find that in texture regions (*e.g.*, dashed box A), the results with texture guidance and double guidance overlap in most circumstances while the structure-guided result shows apparent deviation and oscillations. This is because the semantic structure map is not perfect and still cannot get rid of the negative effect of some strong textures. On the contrary, in regions with

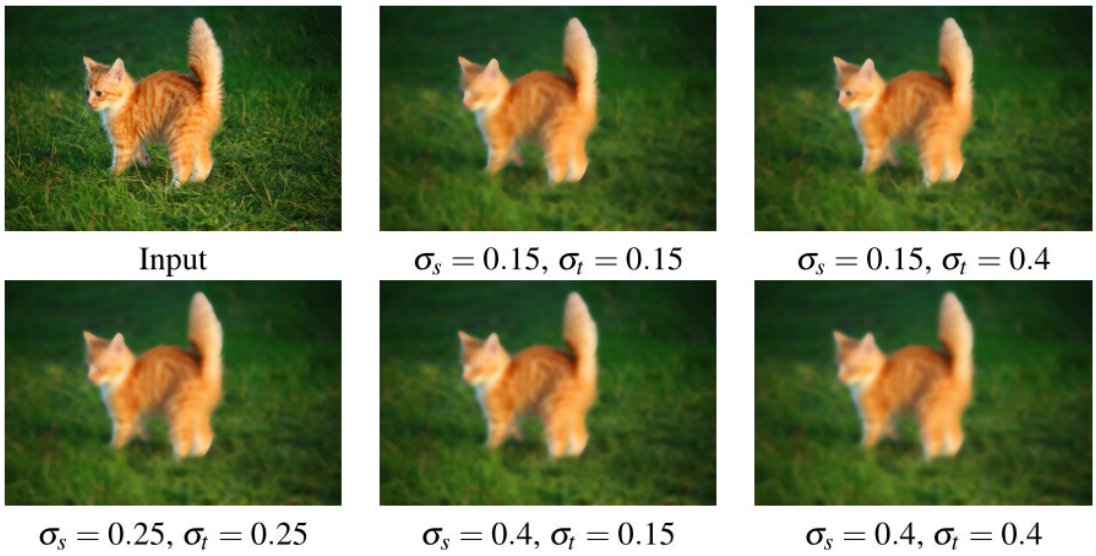



Figure 3.6: Double-guided filtering with different $\sigma_s$ and $\sigma_t$. The two user-specified parameters control the effect of smoothing in terms of structure preservation and texture removal respectively. A smaller $\sigma_s$ can retain more edges and a smaller $\sigma_t$ can smooth out more textures.

edges (*e.g.*, dashed box B), the results with structure guidance and double guidance are almost the same (except the texture regions near edges), indicating that our DGF can properly preserve structures. However, the result of texture guidance in this case is less comparable because edges are over-smoothed (the green line is more rounded and less sharper). From the analysis, we find that structure guidance and texture guidance have different impact on image smoothing, *i.e.*, structure guidance focuses more on structure preservation while texture guidance aims more for texture removal. They are independent in effect but related in the smoothing result. Clearly, the proposed DGF combines the advantages of two guidance effectively.

## 3.4 Experiments

In this section, we demonstrate the effectiveness of the proposed DGF through both visual and quantitative comparison. We also introduce three typical applications of image smoothing at the end.

### 3.4.1 Parameter Adjustment

**Kernel size and iterations.** In our method, the kernel size $k$, and the number of iterations $N_{itr}$, determine the scale of textures to be smoothed and the extent of texture suppression respectively. Fig. 3.5 shows smoothing results with various kernel sizes and iterations to an image with artificial random noise. We examine the signal-to-noise-ratio (SNR) to measure the effect of removing noise quantitatively. Compared

with the noisy input, a smaller SNR indicates that noise is better suppressed. With an increasing kernel size, larger-scale textures are removed more effectively. This can also be achieved by increasing the number of iterations. Empirically, 3-5 iterations with the kernel size of $\{5, 7, 9, 11\}$ can yield desirable results.

**Smoothing effect factors $\sigma_s$ and $\sigma_t$.** The two parameters control the effect of smoothing in terms of preserving structures and removing textures respectively. Normally, a smaller $\sigma_s$ can retain more edges and a smaller $\sigma_t$ can smooth out more textures. Empirically[2]., for good performance, $\sigma_s$ falls into $[0.1, 0.3]$ and $\sigma_t$ into $[0.2, 0.4]$. Fig. 3.6 shows results with various $\sigma_s$ and $\sigma_t$.

### 3.4.2 Comparison with Existing Methods

**Visual comparison.** In Fig. 3.7, we compare our filter with two classical algorithms (total variation (TV) [Rudin et al., 1992], bilateral filter (BLF) [Tomasi and Manduchi, 1998]), and six state-of-the-art algorithms (relative total variation (RTV) [Xu et al., 2012], guided filter (GF) [He et al., 2013], rolling guidance filter (RGF) [Zhang et al., 2014b], fast L0 smoothing [Nguyen and Brown, 2015], segment graph filter (SGF) [Zhang et al., 2015], static and dynamic guidance filter (SDF) [Ham et al., 2015]). Among them, BLF, GF, RGF, SGF are kernel filters, while TV, RTV, fast L0 smoothing, SDF are global approaches. We use the default parameters defined in their open-source codes. In our method, we set $k = 9$, $\sigma_s = 0.15$, $\sigma_t = 0.2$, and $N_{itr} = 3$. With a clearer visualization with close-ups, the DGF outperoforms other methods in suppressing textures more effectively without over-smoothing main structures.

One special and difficult example is the vase image, in which the vase body is covered with very strong textures while the object (vase boundary and base) itself has relatively low contrast to the background. Ideally, the textures should be removed while object-background contrast should be retained. As can be observed, only our DGF removes all the black textures on the vase and preserves weak structures of the vase simultaneously. Other methods cannot achieve both goals. Even though in some cases, *e.g.*, the results produced by TV, RTV and SGF, textures are suppressed somewhat. However, the base-background contrast is completely lost as a side-effect. This example further shows the superior performance of our method in preserving main structures and removing textures.

**Quantitative evaluation.** Since denoising is a basic function of image smoothing, we can further evaluate the denoising performance with SNR quantitatively, similar to [Zhang et al., 2015; Liu et al., 2013b; Zhang et al., 2014b; Nguyen and Brown, 2015; Yang, 2016]. More specifically, we first take a smoothed image[3] as ground truth (original signal), and then add Gaussian noise with the standard deviation as 0.05. The SNR here measures the effect of removing noise (compared with ground truth, a larger SNR indicates that the noise is better removed). We show three groups of

[2]We determine the range of $\sigma_s$ and $\sigma_t$ on visual results by looking at 100 natural images from the Internet. Some of them are displayed in Fig. 3.5, Fig. 3.6, Fig. 3.7, Fig. 3.8, and Fig. 3.9

[3]The smoothed image is generated by manually tuning the parameters of RTV [Xu et al., 2012] to produce the most human-pleasing results.

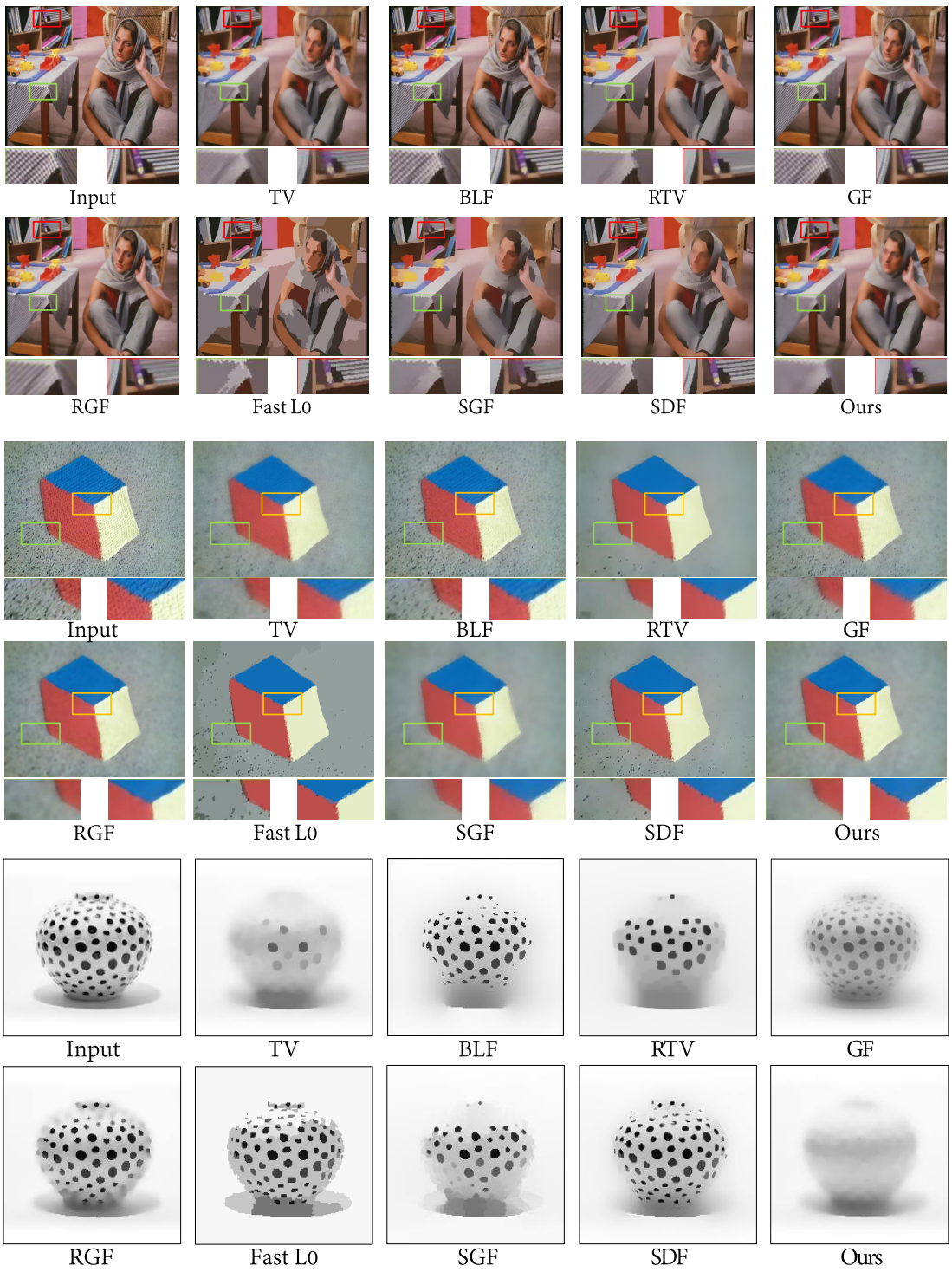



Figure 3.7: Comparison of image smoothing results with different methods. The methods we compare include TV [Rudin et al., 1992], BLF [Tomasi and Manduchi, 1998], RTV [Xu et al., 2012], GF [He et al., 2013], RGF [Zhang et al., 2014b], Fast L0 [Nguyen and Brown, 2015], SGF [Zhang et al., 2015], and SDF [Ham et al., 2015]. Our DGF consistently performs better in preserving structures and removing textures.

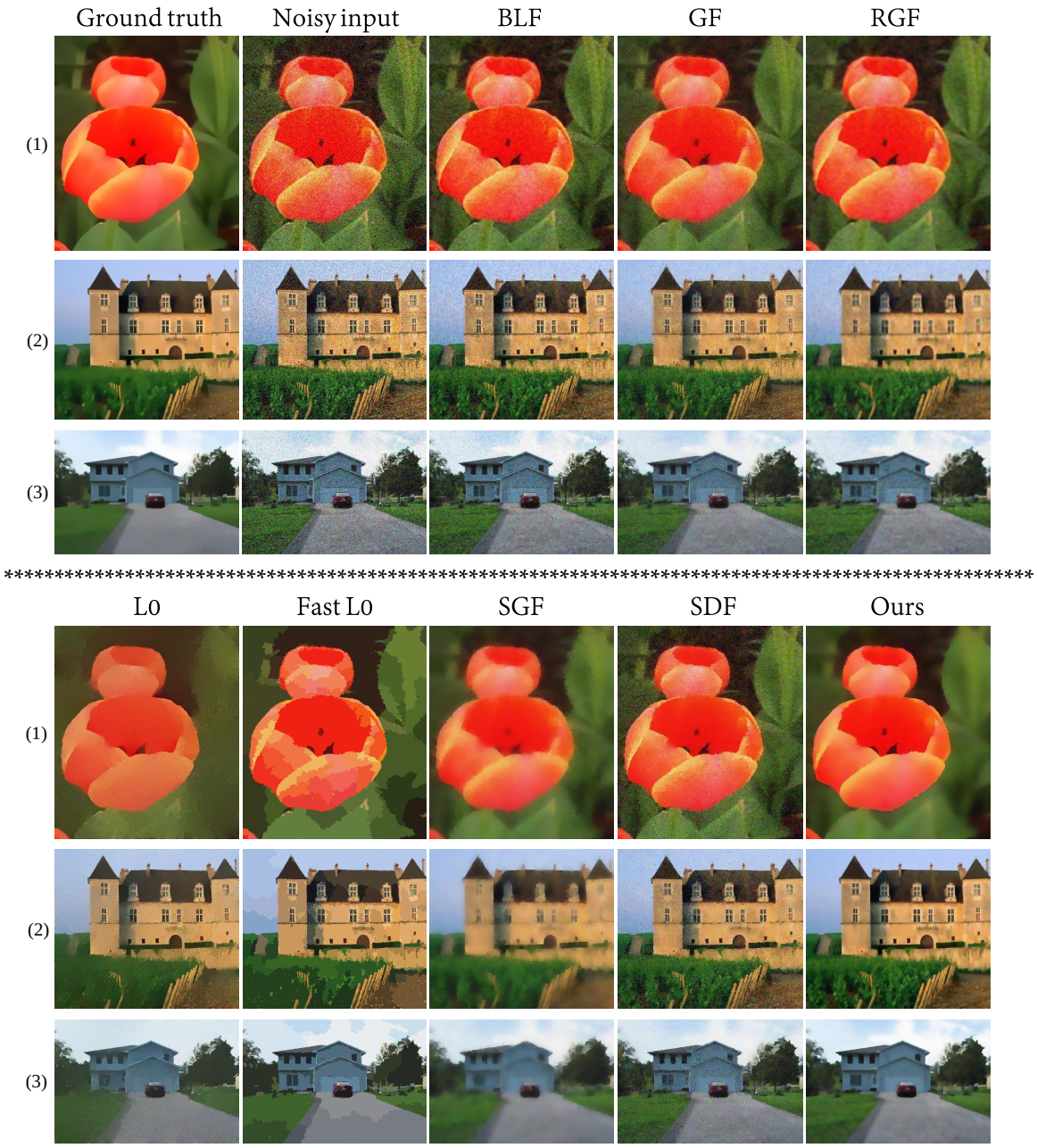
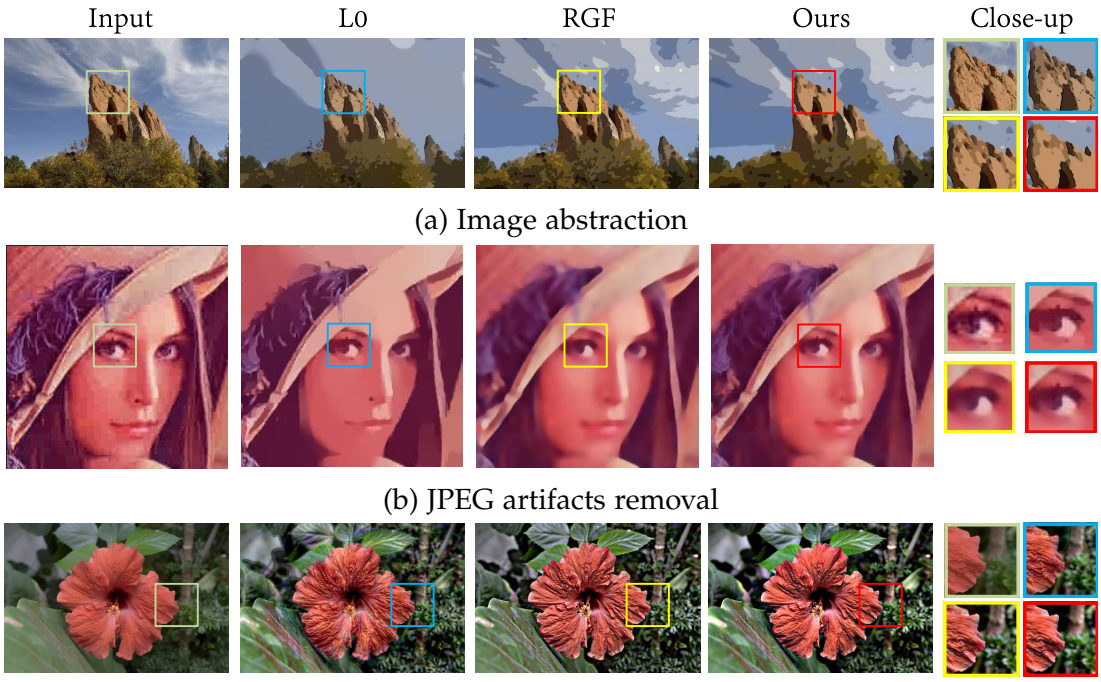



Figure 3.8: Image denoising results with different methods. The methods we compare include BLF [Tomasi and Manduchi, 1998], GF [He et al., 2013], RGF [Zhang et al., 2014b], L0 [Xu et al., 2011], Fast L0 [Nguyen and Brown, 2015], SGF [Zhang et al., 2015], and SDF [Ham et al., 2015]. Our DGF has consistently better denoising performance.




(a) Image abstraction

(b) JPEG artifacts removal

(c) Detail enhancement

Figure 3.9: Image smoothing applications. The methods we compare are L0 [Xu et al., 2011] and RGF [Zhang et al., 2014b]. Our DGF consistently outperforms the two methods in producing better visual results.

results in Fig. 3.8, and list corresponding SNR values in Table 3.1. It is clear that SNR values of our filter are the best in all the three examples, showing that our method can suppress noise better. Moreover, our results are visually closer to ground truth images. It should be noted that although RGF [Zhang et al., 2014b] and Fast L0 [Nguyen and Brown, 2015] can both yield relatively large SNR, their visual results are less competitive. RGF makes the output look blurry, especially around edges and corners. Fast L0 introduces more noticeable quantization artifacts into results.

### 3.4.3 Applications

**Image abstraction.** Image abstraction aims to create a cartoon-like style from an input image. We use the method in [Winnemöller et al., 2006] for image abstraction. The results of a mountain are shown in Fig. 3.9(a). It is clear that our method can suppress more details on the surface of the mountain while well preserving its main structure.

**JPEG artifacts removal.** The quality of JPEG compression images are always degraded by unwanted artifacts, which can be removed by image smoothing algorithms. The results are shown in Fig. 3.9(b). We observe that our method removes artifacts more effectively than the other two methods.

**Detail enhancement.** Suppose $I$ is the input image, and $S$ is the smoothed output.

| Method | Fig. 3.8(1) | Fig. 3.8(2) | Fig. 3.8(3) | Average |
|---|---|---|---|---|
| Noisy input | 30.99 | 33.12 | 36.06 | 33.39 |
| BLF [Tomasi and Manduchi, 1998] | 45.73 | 44.54 | 48.28 | 46.18 |
| GF [He et al., 2013] | 44.77 | 42.95 | 47.59 | 45.10 |
| RGF [Zhang et al., 2014b] | 54.45 | 49.59 | 56.20 | 53.41 |
| L0 [Xu et al., 2011] | 35.82 | 45.37 | 48.14 | 43.11 |
| Fast L0 [Nguyen and Brown, 2015] | 47.37 | 47.10 | 51.73 | 48.73 |
| SGF [Zhang et al., 2015] | 50.63 | 42.25 | 50.45 | 47.78 |
| SDF [Ham et al., 2015] | 41.39 | 42.01 | 46.82 | 43.41 |
| Ours | **58.36** | **63.25** | **62.69** | **61.43** |

Table 3.1: SNR values of images in Fig. 3.8. Our DGF achieves the best quantitative results in all the three examples.

We define detail enhancement $DE$ as: $DE = S + \alpha \cdot (I - S)$, where $\alpha \geq 1$ controls the extent ($\alpha = 2$ in our case). The results with different methods are shown in Fig. 3.9(c). With close inspection of some texture regions, our method performs better in boosting the details without affecting the overall color tone and over-boosting edges.

## 3.5 Conclusion

In this chapter, we have proposed the double-guided filter (DGF) that utilizes structure guidance and texture guidance simultaneously. As a primary novelty, we have introduced the concept of texture guidance which fundamentally improves traditional kernel-based methods in differentiating between structures and textures more effectively. The combination of structure guidance and texture guidance makes the filter both "structure-aware" and "texture-aware". The proposed DGF outperforms existing image smoothing methods in preserving main structures and removing insignificant textures. Extensive experiments have demonstrated the effectiveness of the DGF. Our future work will focus on implementing new methods for constructing structure guidance and texture guidance, and accelerating the filtering process with GPU parallel computing.

Chapter 4

# Texture and Structure Aware Filtering Network for Image Smoothing

In the previous chapter, we proposed a kernel-based double-guided filter (DGF) which leverages structure guidance and texture guidance at the same time. Structure guidance comes from a semantic edge detection method, which is beneficial for preserving more semantically meaningful structures. As a primary novelty, we introduced the concept of "texture guidance" that indicates the position and magnitude of textures with a confidence map. It is obtained by normalizing the texture layer of a global method. Double guidance equips the proposed filter with both "structure-awareness" and "texture-awareness", and improves image smoothing in removing strong textures without degrading main structures.

In this chapter, we push structure guidance, texture guidance, and image smoothing further by leveraging deep neural networks. The motivation is that hand-crafted features cannot appropriately and robustly identify and extract natural textures as they present high randomness in appearance, *i.e.*, spatial distortion and color variations. Although several deep smoothing networks have been proposed for extracting richer image features, they mainly approximate existing hand-crafted filters by using their output as ground truth. Hence, they still cannot overcome the shortcomings of these filters in differentiating between structures and textures. To deal with this fundamental problem, we generate synthetic data by blending natural textures with clean structure-only images. With the data, we build a texture prediction network (TPN) that estimates the location and magnitude of textures, *i.e.*, texture guidance. We additionally take advantage of a semantic structure prediction network (SPN) to generate structure guidance. We then incorporate the two forms of guidance into the filtering network that constitutes our texture and structure aware filtering network (TSAFN). TSAFN is able to more effectively identify the textures to remove ("texture-awareness") and the structures to preserve ("structure-awareness"). Experimental results demonstrate that the proposed model achieves superior performance in texture prediction and image smoothing, and generalizes well to natural images.

In the reminder of this chapter, Section 4.1 introduces our motivation and summa-

rizes primary contributions. Section 4.2 illustrates the properties of natural textures and the process of blending textures with structure-only images, and depicts the texture prediction network. We incorporate texture guidance and semantic structure guidance into the deep filtering network in Section 4.3. Section 4.4 validates the effectiveness of the proposed deep filtering model through qualitative and quantitative results. Section 4.5 summarizes the chapter and proposes the future work.

## 4.1 Introduction

Image smoothing, a fundamental technology in image processing and computer vision, aims to enhance images by retaining salient structures (to the *structure-only image*) and removing insignificant textures (to the *texture-only image*).

There are mainly two types of methods for image smoothing: (1) kernel-based methods, that calculate the weighted average of neighbouring pixels, such as the guided filter (GF) [He et al., 2013], rolling guidance filter (RGF) [Zhang et al., 2014b], segment graph filter (SGF) [Zhang et al., 2015]; and (2) separation-based methods, which decompose the image into a structure layer and a texture layer, such as relative total variation (RTV) [Xu et al., 2012], fast L0 [Nguyen and Brown, 2015], and static and dynamic guidance filter (SDF) [Ham et al., 2017]. These approaches rely on hand-crafted features to distinguish textures from structures, which are largely based on low-level appearance. They generally assume that structures always have larger gradients, while textures are just smaller oscillations in color intensities.

In fact, it is quite difficult to identify and extract textures. The main reasons are twofold: (1) Textures are essentially repeated patterns regularly or irregularly distributed within object structures, and they may present significant spatial distortion and color variations (Fig. 4.1(a) and Fig. 1.3), making it hard to explicitly define and model them mathematically; (2) There are strong textures with large gradients and color contrast to the background in some images, which are easy to confuse with structures (such as white stripes on the giraffe's body in Fig. 4.1(c)). We see from Fig. 4.1 that GF, RGF, SGF, fast L0, and SDF perform poorly on the giraffe image. The textures are either not removed, or suppressed with the structure severely blurred. This is because the hand-crafted nature of these filters makes them less robust when applied to various types of textures, which leads to poor discrimination of textures and structures. Some other methods [Xu et al., 2015; Liu et al., 2016; Li et al., 2016; Fan et al., 2017b; Chen et al., 2017a; Fan et al., 2017a; Shen et al., 2017] take advantage of deep neural networks, and aim for improving the performance by extracting richer features. However, existing networks use the output of various hand-crafted filters as ground truth during training. These deep learning approaches are thus limited by the shortcomings of these hand-crafted filters, and cannot learn how to effectively differentiate between textures and structures.

The double-guided filter (DGF) [Lu et al., 2017] proposed in the previous chapter addresses this issue by introducing the idea of "texture guidance", which infers the location and magnitude of textures, and combines it with "structure guidance" to

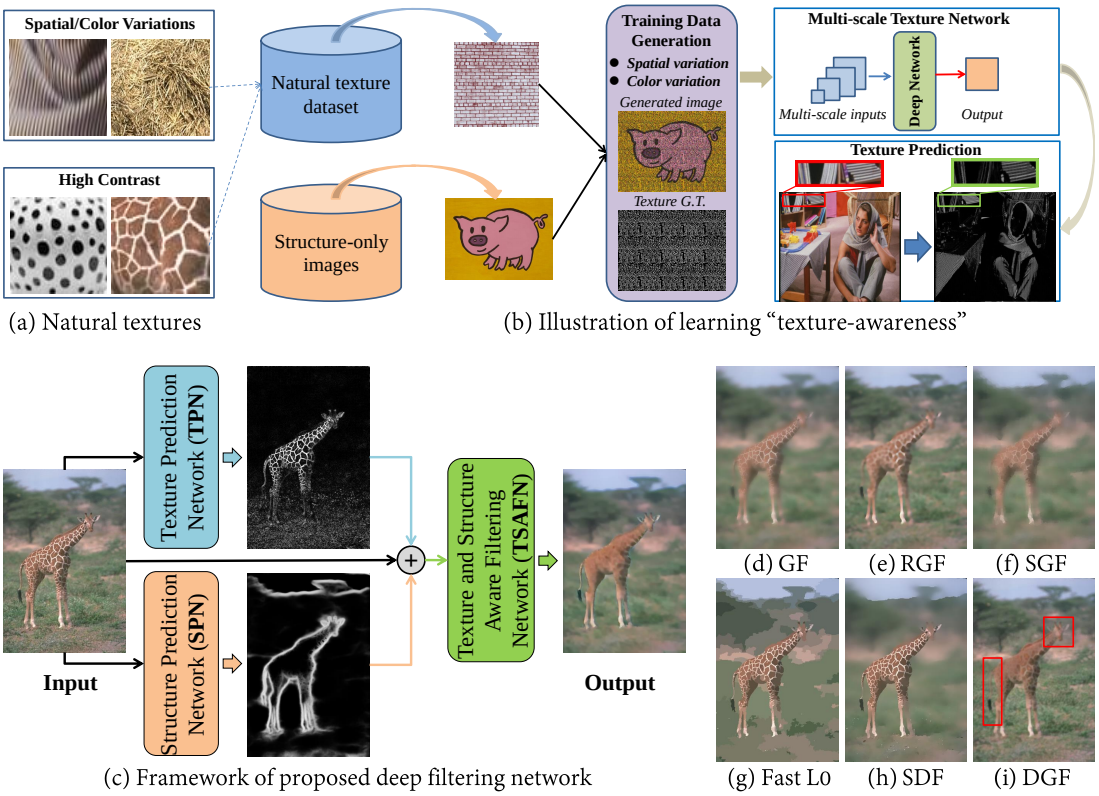



Figure 4.1: (a) Texture in natural images is often hard to identify due to spatial distortion and high contrast. (b) Illustration of learning "texture awareness". We generate training data by adding spatial and color variations to natural texture patterns and blending them with structure-only images, and then use the result to train a multi-scale texture network with texture ground-truth. We test the network on both generated data and natural images. (c) The proposed deep filtering network is composed of a texture prediction network (TPN) for predicting textures (white stripes with high-contrast); a structure prediction network (SPN) for extracting structures (the giraffe's boundary, which has relatively low contrast to the background); and a texture and structure aware filtering network (TSAFN) for image smoothing. (d)-(i) Existing methods, *e.g.*, GF [He et al., 2013], RGF [Zhang et al., 2014b], SGF [Zhang et al., 2015], Fast L0 [Nguyen and Brown, 2015], SDF [Ham et al., 2017], cannot distinguish high-contrast textures from structures effectively.

achieve both goals of texture removal and structure preservation. However, the DGF uses a hand-crafted separation-based algorithm called Structure Gradient and Texture Decorrelating (SGTD) [Liu et al., 2013b] to construct the texture confidence map that still cannot essentially overcome the natural deficiency. We argue that this is not true "texture awareness", because in many cases, some structures are still blurred when the filter tries to remove strong textures after several iterations. As can be seen in Fig. 4.1(i), although the stripe textures are mostly smoothed out, the giraffe's structure is severely blurred, especially around the head and the tail (red boxes).

In this chapter, we hold the idea that “texture awareness" should reflect both the *texture region* (where the texture is) and *texture magnitude* (texture with high contrast to the background is harder to remove). Thus, we take advantage of deep learning and propose a texture prediction network (TPN) that aims to learn textures from natural images. However, since there are no available datasets containing natural images with labelled texture regions, we make use of texture-only datasets [Cimpoi et al., 2014; Dana et al., 1999]. The process of learning “texture awareness" is shown in Fig. 4.1(b). Specifically, we generate the training data by adding spatial and color variations to natural texture patterns and blending them with structure-only images. Then we construct a multi-scale network (containing different levels of contextual information) to train these images with texture ground truth. The proposed TPN is able to predict textures through a full consideration of both low-level appearance, *e.g.*, gradients, and other statistics, *e.g.*, repetition, tiling, spatial varying distortion. In both our synthetic data and natural images, the network achieves good performance on locating texture regions and measuring texture magnitude by assigning different confidence values, as shown in Fig. 4.1(b).

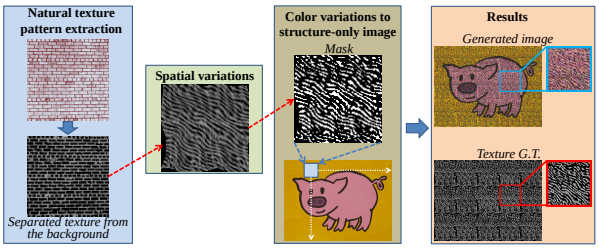
For the full problem, we are inspired by the idea of “double guidance" introduced in [Lu et al., 2017] and propose a deep neural network based filter that learns to predict textures to remove (“texture-awareness" by our TPN) and structures to preserve (“structure-awareness" by HED semantic edge detection [Xie and Tu, 2015]). This is an end-to-end image smoothing architecture which we refer to as “Texture and Structure Aware Filtering Network" (TSAFN), as shown in Fig. 4.1(c). The network is trained with our synthetic data. Different from the DGF [Lu et al., 2017], we generate texture guidance and structure guidance with deep learning approaches, and replace the hand-crafted kernel filter with the model to achieve a more effective combination of the forms of guidance. Experimental results show that the proposed deep filter outperforms the DGF [Lu et al., 2017] and other state-of-the-art smoothing methods significantly on synthetic and natural images.

In summary, we make the following major contributions:

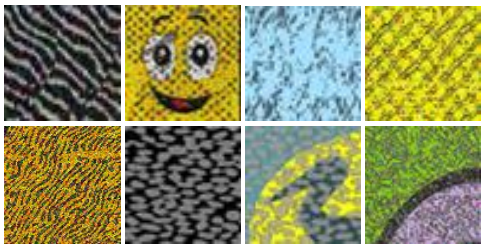
- We give more theoretical insight into textures and propose a deep neural network to robustly predict textures in natural images.
- We present synthetic data that enable the training of texture prediction and image smoothing. It also allows to objectively evaluate smoothing results via quantitative comparison.
- We propose an end-to-end deep neural network for image smoothing that achieves both “texture-awareness" and “structure-awareness", and outperforms existing methods on challenging natural images.

## 4.2 Texture Prediction Network

In this section, we give insights into textures in natural images, which inspire the design of the texture prediction network (TPN) and the synthetic data for training.




(a) Data generation process



(b) More examples of generated data

Figure 4.2: Illustration of synthetic data generation. (a) We blend natural texture patterns with structure-only images, adding spatial and color variations to increase texture diversity. (b) We show more examples of generated data, and textures there present various levels of contrast to the background.

### 4.2.1 Textures in Natural Images

Generally, textures are repetitive patterns regularly or irregularly distributed within object structures [Yalniz and Aksoy, 2010; Cai and Baciu, 2013]. Texture appearance in natural images presents random and diverse variations in spatial arrangement and color intensities (see Fig. 1.3), *i.e.*, spatial distortion and color variations are two essential properties of natural textures. In addition to the inherent appearance of textures, these variations may also be related to the object surface property and camera conditions. In the natural world, object surfaces always have different orientations caused by their underlying curvature. After projected to an image with perspective distortion, texture appearance presents significant variations in size, density, placement, and color. However, it is unrealistic to represent object surfaces uniformly and record the camera condition of every image. Considering all these factors, it is difficult for hand-crafted features, *i.e.*, low-level cues like intensity difference or gradients, to robustly model and identify textures, especially in natural images.

To tackle these issues, we take advantage of deep networks that can extract richer image features and generalize better to randomness of image patterns without explicit modelling [Krizhevsky et al., 2012; He et al., 2016]. Provided sufficient training examples are available, the network is able to learn to predict textures more robustly.

### 4.2.2 Data Generation

We aim to provide training data so that the deep network can learn to predict textures. Ideally, we would like to learn directly from natural images. However, manually annotating pixel-wise labels would be costly. Moreover, it would require a full range of textures, each with a full range of distortion in a broad array of natural scenes. Labelling them becomes extremely difficult. Therefore, we propose an approach to generating synthetic training data. Later, we will demonstrate that the texture prediction network can effectively predict natural textures with our data.

We observe that cartoon images have only structural edges filled with pure color, and can be safely considered as “structure-only images". Specifically, we select 174

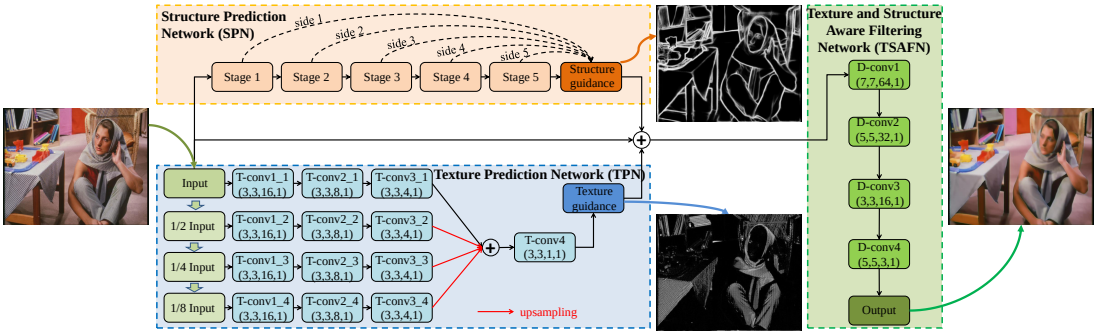



Figure 4.3: The proposed network architecture. The outputs of the texture prediction network (TPN) and structure prediction network (SPN) are concatenated with the original input, and then fed to the texture and structure aware filtering network (TSAFN) to produce the final smoothing result. (*k*,*k*,*c*,*s*) for a convolutional layer means the kernel is $k \times k$ in size with $c$ feature maps, and the stride is $s$.

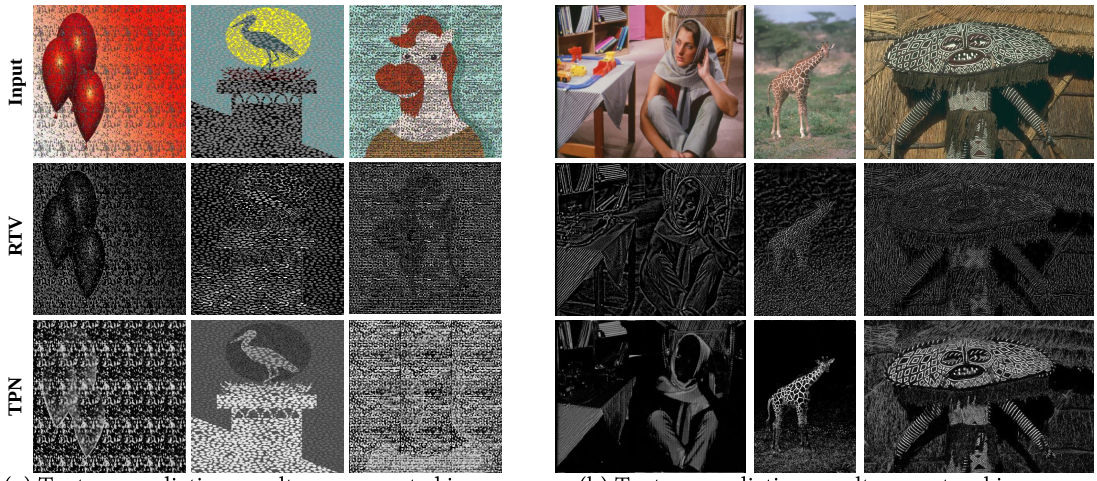
cartoon images from the Internet and 233 different types of natural textures from public datasets [Cimpoi et al., 2014; Dana et al., 1999]. The data generation process is illustrated in Fig. 4.2(a). Note that texture images in these datasets only contain pure textures and most of them have regular patterns, so separating them from the background is quite simple even with a hand-crafted method. We use Relative Total Variation (RTV) [Xu et al., 2012] for its efficiency and flexible parameter tuning. We obtain textures by normalizing the texture layer from RTV into $[0, 1]$.

To mimic natural textures, we employ both spatial and color variations to aforementioned RTV-extracted textures during data generation. As shown in Fig. 4.2(a), we blend textures with structure-only images. In detail, we first rescale normalized texture images from RTV to $100 \times 100$. For spatial variations, we perform geometric transformations including rotation, scaling, shearing, and some irregular and random pixel displacement. We randomly select parameters for these random transformations. Based on the deformed result, we generate a binary mask $\mathbf{M}$, *i.e.*, values larger than 0.2 are updated to 1.

As for color variations, given the structure-only image $\mathbf{S}$, the value of pixel $i$ in the $j^{th}$ channel of the generated image $\mathbf{I}_i^{(j)}$ is determined by both $\mathbf{S}$ and the mask $\mathbf{M}$. If $\mathbf{M}_i = 1$, $\mathbf{I}_i^{(j)} = rand[\kappa \cdot (1 - \mathbf{S}_i^{(j)}), 1 - \mathbf{S}_i^{(j)}]$, where $\kappa$ is used to control the range of random generation and empirically set as 0.75. Otherwise, $\mathbf{I}_i^{(j)} = \mathbf{S}_i^{(j)}$. We repeat this by sliding the mask over the whole image without overlapping. Ground truth texture confidence is calculated by averaging the values of the three channels of the texture layer:

$$\mathbf{T}_i^* = \delta\left(\frac{1}{3}\sum_{j=1}^{3}\left|\mathbf{I}_i^{(j)} - \mathbf{S}_i^{(j)}\right|\right), \tag{4.1}$$

where $\delta(x) = 1/(1 + exp(-x))$ is the sigmoid function to scale the value to $[0, 1]$. We use color variations to generate various levels of contrast between textures and the




(a) Texture prediction results on generated images (b) Texture prediction results on natural images

Figure 4.4: Texture prediction results. First row: input (including both generated and natural images). Second row: texture extraction results by RTV [Xu et al., 2012]. Third row: texture prediction results by the proposed TPN. TPN is able to localize textures in both generated and natural images effectively, and indicate the magnitude of textures by assigning pixel-level confidence. By contrast, RTV cannot appropriately extract textures.

background (most color contrast is large, *i.e.*, corresponding to strong textures). With this approach, it is less likely that two images have identical (or very similar) texture patterns even when the textures come from the same original pattern. Fig. 4.2(b) shows some examples of generated images.

Finally, we generate 30,000 images in total (a handful of low-quality images have been manually removed). For ground truth, besides purely-clean structure-only images, we also provide binary structure maps and texture confidence maps of all the synthetic data. Currently we distribute textures over the entire image and the textures are not object-dependent, which may not be typical appearance in natural images. To bridge the gap, in addition to utilizing natural textures and increasing their variations as illustrated above, we also use patch learning, *i.e.*, training the network on image patches. This is motivated by the fact that textures are always clustered in local regions. Moreover, we aim to let the network learn the statistics of the appearance of local textures rather than their global structures. Transfer learning [Dai et al., 2007; Noroozi et al., 2018] can be employed to further enhance texture adaptation in our future work.

### 4.2.3 Network Architecture

**Network design.** We propose the texture prediction network (TPN), which is trained on our generated data. Considering that textures have various colors, scales, and shapes, we employ a multi-scale learning strategy. Specifically, we apply 1/2, 1/4, and 1/8 down-sampling to the input respectively. For each image, we use 3 convo-

lutional layers for feature extraction, with the same size $3 \times 3$ kernel but different numbers of feature maps. Then, all the feature maps are resized to the original input size and concatenated to form a 16-channel feature map. They are further convolved with a $3 \times 3$ layer to yield the final 1-channel result. Note that each convolutional layer is followed by ReLU except for the output layer. The output layer uses the sigmoid activation function to scale values to $[0, 1]$. The architecture of TPN is shown in Fig. 4.3. Consequently, given the input image $\mathbf{I}$, the predicted texture guidance $\tilde{\mathbf{T}}$ is obtained by:

$$\tilde{\mathbf{T}} = TPN\left(\mathbf{I}, \frac{1}{2}\mathbf{I}, \frac{1}{4}\mathbf{I}, \frac{1}{8}\mathbf{I}\right). \tag{4.2}$$

**Loss function.** The network is trained by minimizing the mean squared error (MSE) between the predicted texture guidance map and ground truth:

$$\ell_T = \frac{1}{N}\sum_i \left\|\tilde{\mathbf{T}}_i - \mathbf{T}_i^*\right\|_2^2, \tag{4.3}$$

where $N$ is the number of pixels in the image, $*$ denotes ground truth data. More training details can be found in Section 4.4.1.

**Texture prediction results.** We display texture prediction results on our generated images in Fig. 4.4(a) and natural images in Fig. 4.4(b). The network is able to localize textures in both generated and natural images effectively, and indicate the magnitude of textures by assigning pixel-level confidence (the third row). For comparison, we also visualize texture extraction results from these examples by RTV [Xu et al., 2012] in the second row. RTV performs less competitively on these complex scenes. This is because just like other hand-crafted filters, RTV also assumes structures have large gradients, so it has poor discrimination of strong textures. Hence, in RTV results, most strong textures are not properly extracted, and some structure components are incorrectly included in the texture map.

## 4.3 Texture and Structure Aware Filtering Network

As shown in Fig. 4.3, our deep filtering network consists of three parts:

1. Texture prediction network **TPN**, that constructs texture guidance to indicate texture regions and magnitude (texture confidence).

2. Structure prediction network **SPN**, that constructs structure guidance to indicate meaningful structures (structure confidence).

3. Texture and structure aware filtering network **TSAFN**, that concatenates texture guidance and structure guidance with the original input and generates the smoothed output.

Since TPN has been discussed in the previous section, we give more details on SPN and TSAFN in the following.

### 4.3.1 Structure Prediction Network

Structure information is an essential cue for image smoothing, which indicates the structures to be preserved. Ideal structure guidance should give high confidence to meaningful structures, regardless of their gradients. We utilize a recently-proposed holistically-nested edge detection (HED) [Xie and Tu, 2015] as the structure prediction network (SPN):

$$\tilde{\mathbf{E}} = HED(\mathbf{I}) = \text{fuse}(\tilde{\mathbf{E}}^{(1)},...,\tilde{\mathbf{E}}^{(5)}), \tag{4.4}$$

where $\tilde{\mathbf{E}}^{(m)}$ is the side output from the $m^{th}$ stage (each stage contains several convolutional and pooling layers). The final loss is denoted as $\ell_E$. Please refer to the original paper [Xie and Tu, 2015] for more details.

### 4.3.2 Deep Filtering Network

Once structure guidance and texture guidance are generated, the texture and structure aware filtering network (TSAFN) concatenates them with the input to form a 5-channel tensor. TSAFN consists of 4 layers. We set a relatively large kernel ($7 \times 7$) in the first layer to contain more original information. The kernel size decreases in the following two layers ($5 \times 5$, $3 \times 3$ respectively). In the last layer, the kernel size is increased to $5 \times 5$ again. The first three layers are followed by ReLU, while the last layer has no activation function (transforming the tensor into the 3-channel output). Empirically, we remove all pooling layers, the same as [Xu et al., 2015; Li et al., 2016; Fan et al., 2017b; Chen et al., 2017a]. The whole process can be denoted as:

$$\tilde{\mathbf{I}} = TSAFN(\mathbf{I}, \tilde{\mathbf{E}}, \tilde{\mathbf{T}}). \tag{4.5}$$

The network is trained by minimizing:

$$\ell_D = \frac{1}{N} \sum_i (\left\| \tilde{\mathbf{I}}_i - \mathbf{I}_i^* \right\|_2^2). \tag{4.6}$$

where $\mathbf{I}^*$ is the structure-only image that is used as ground truth. Note that the $\ell_2$ loss is widely employed in image smoothing [Xu et al., 2015; Li et al., 2016; Fan et al., 2017b], and has been proved to be effective in producing higher accuracy in quantitative evaluation [Chen et al., 2017a] and facilitating convergence [Zhao et al., 2016].

## 4.4 Experiments

In this section, we demonstrate the effectiveness of the proposed deep image smoothing network through extensive experiments.

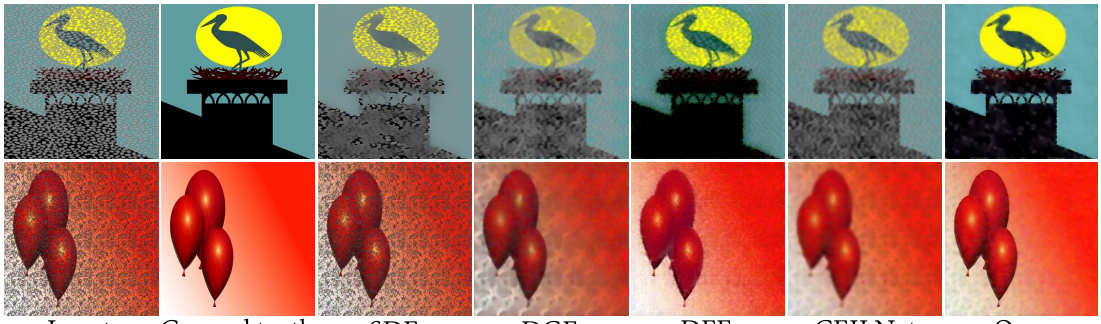



Figure 4.5: Smoothing results on generated images. Our filter can smooth out various types of textures while preserving main structures more effectively than other approaches, *i.e.*, SDF [Ham et al., 2017], DGF [Lu et al., 2017], DFF [Chen et al., 2017a], and CEILNet [Fan et al., 2017b] (DFF and CEILNet are trained on our data).

### 4.4.1 Implementation Details

**Environment setup.** We construct networks in Tensorflow [Abadi et al., 2016], and train and test all the data on a single NVIDIA Titan X graphics card.

**Dataset.** Because there is no publicly-available image smoothing dataset, we perform training using our generated images. More specifically, we select 19,505 images (65%) from synthetic data for training, 2,998 (10%) for validation, and 7,497 (25%) for testing (all test images are resized to $512 \times 512$). There is no overlapping of structure-only and texture images between training, validation and testing samples.

**Training procudure.** We first train the three networks separately. 300,000 patches with the size $64 \times 64$ are randomly and sparsely collected from training images. We use gradient descent with a learning rate of $10^{-4}$, and momentum of 0.9. Finally, we perform fine-tuning by jointly training the entire network with a smaller learning rate of $10^{-5}$, and the same momentum 0.9. The fine-tuning loss is

$$\ell_F = \gamma \cdot \ell_D + \lambda \cdot (\ell_T + \ell_E), \tag{4.7}$$

where we empirically set $\gamma = 0.6$ and $\lambda = 0.2$ according to the validation set.

### 4.4.2 Comparison with Existing Methods

**Methods to compare.** For non-learning methods, we compare the proposed method with two classical filters: Total Variation (TV) [Rudin et al., 1992], bilateral filter (BLF) [Tomasi and Manduchi, 1998], and recently-proposed filters: L0 [Xu et al., 2011], Relative Total Variation (RTV) [Xu et al., 2012], guided filter (GF) [He et al., 2013], Structure Gradient and Texture Decorrelation (SGTD) [Liu et al., 2013b], rolling guidance filter (RGF) [Zhang et al., 2014b], fast L0 [Nguyen and Brown, 2015], segment graph filter (SGF) [Zhang et al., 2015], static and dynamic filter (SDF) [Ham et al., 2017], double-guided filter (DGF) [Lu et al., 2017] proposed in Chapter 3. Note that, BLF, GF, RGF, SGF, DGF are kernel-based, while TV, L0, RTV, SGTD, fast L0, SDF are

separation-based. We use default parameters defined in their open-source codes for each method.

We additionally compare five state-of-the-art deep filtering models: deep edge-aware filter (DEAF) [Xu et al., 2015], deep joint filter (DJF) [Li et al., 2016], deep recursive filter (DRF) [Liu et al., 2016], deep fast filter (DFF) [Chen et al., 2017a], and cascaded edge and image learning network (CEILNet) [Fan et al., 2017b] . **We retrain all the models with our data**.

| | PSNR ↑ | SSIM ↑ | Time ↓ | | PSNR ↑ | SSIM ↑ | Time ↓ |
|---|---|---|---|---|---|---|---|
| TV | 11.33 | 0.6817 | 2.44 | RGF | 15.73 | 0.7173 | 0.87 |
| BLF | 10.89 | 0.6109 | 4.31 | Fast L0 | 15.50 | 0.7359 | 1.36 |
| L0 | 14.62 | 0.7133 | 0.94 | SGF | 13.92 | 0.7002 | 2.26 |
| RTV | 14.07 | 0.7239 | 1.23 | SDF | 16.82 | 0.7633 | 3.71 |
| GF | 12.22 | 0.6948 | 0.83 | DGF | 17.89 | 0.7552 | 8.66 |
| SGTD | 16.14 | 0.7538 | 1.59 | Ours | **25.07** | **0.9152** | **0.16** |

Table 4.1: Quantitative evaluation of different non-learning filters tested on our test data. The methods we compare include TV [Rudin et al., 1992], BLF [Tomasi and Manduchi, 1998], L0 [Xu et al., 2011], RTV [Xu et al., 2012], GF [He et al., 2013], SGTD [Liu et al., 2013b], RGF [Zhang et al., 2014b], fast L0 [Nguyen and Brown, 2015], SGF [Zhang et al., 2015], SDF [Ham et al., 2017], and DGF [Lu et al., 2017]. ↑ means larger is better, and ↓ means smaller is better. The best results are marked as **bold**.

**Results on generated images.** We first compare the average PSNR [Hore and Ziou, 2010], SSIM [Wang et al., 2004], and processing time (in seconds) of non-learning filters on our testing data in Table 4.1. Our method achieves the the best PSNR (closer to ground truth) and SSIM (more accurately preserving structures), and lowest running time, indicating its superiority in both effectiveness and efficiency.

We also compare the quantitative results on different deep models trained and tested on our synthetic data in Table 4.2. Our model achieves the best PSNR and SSIM with comparable efficiency. We additionally select 4 state-of-the-art methods (SDF [Ham et al., 2017], DGF [Lu et al., 2017], DFF [Chen et al., 2017a], and CEILNet [Fan et al., 2017b]) for visual comparison in Fig. 4.5. The textures in the first example have relatively large scale. SDF, DGF, and CEILNet attempt to remove these textures but the structures are severely blurred as a side effect. In the second example, the textures are densely distributed and have relatively large contrast. SDF does not have good performance in this example due to the poor texture discrimination. DGF and CEILNet can suppress these textures, but the structures are blurred. Although DFF can smooth out almost all the textures, the final results present unexpected artifacts and color shift, and look less similar to ground truth. Our filter performs consistently well in both examples. We provide more qualitative results on synthetic data in Fig. 4.9.

| | PSNR ↑ | SSIM ↑ | Time ↓ | | PSNR ↑ | SSIM ↑ | Time ↓ |
|---|---|---|---|---|---|---|---|
| DEAF | 20.62 | 0.8071 | 0.35 | DFF | 22.21 | 0.8675 | **0.07** |
| DJF | 19.01 | 0.7884 | 0.28 | CEILNet | 22.65 | 0.8712 | 0.13 |
| DRF | 21.14 | 0.8263 | 0.12 | Ours | **25.07** | **0.9152** | 0.16 |

Table 4.2: Quantitative evaluation of deep models trained and tested on our data. The methods we compare include DEAF [Xu et al., 2015], DJF [Li et al., 2016], DRF [Liu et al., 2016], DFF [Chen et al., 2017a], CEILNet [Fan et al., 2017b]. ↑ means larger is better, and ↓ means smaller is better. The best results are marked as **bold**.

| | PSNR ↑ | SSIM ↑ |
|---|---|---|
| No guidance (Baseline) | 20.32 | 0.7934 |
| Only structure guidance | 21.71 | 0.8671 |
| Only texture guidance | 23.23 | 0.8201 |
| Two guidance (trained separately) | 24.78 | 0.9078 |
| Two guidance (fine-tuned) | **25.07** | **0.9152** |

Table 4.3: Ablation study of image smoothing results with no guidance, only structure guidance, only texture guidance, and two guidance (trained separately and fine-tuned). ↑ means larger is better. The best results are marked as **bold**.

**Results on natural images.** We visually compare smoothing results on five challenging natural images[1] with SDF [Ham et al., 2017], DGF [Lu et al., 2017], DFF [Chen et al., 2017a], and CEILNet [Fan et al., 2017b] in Fig. 4.6. In the first example, the leopard is covered with black textures, and its structure is relatively weak, *i.e.*, it has low contrast to the background. Only our filter smooths out all the textures while effectively preserving the structure. The next four examples present various texture types with different shapes, contrast, and distortion. Our filter performs consistently well in all the examples. We analyze the last challenging vase example in more detail. The vase is covered with strong dotted textures, which are densely distributed on the surface. SDF fails to remove these textures since they are regarded as structures with large gradients. DGF smooths out more black dots but the entire image looks blurry. This is because a larger kernel size and more iterations are required to remove more textures, which inevitably leads to blurred structures. The two deep filters do not demonstrate much improvement over the hand-crafted approaches because "texture-awareness" is not specially addressed in their network design. Our filter outperforms these methods in removing more textures (including strong textures) without degrading main structures. We provide more qualitative results on natural images in Fig. 4.10.

[1]These natural images are from the BSDS dataset [Arbelaez et al., 2010] and the Internet.

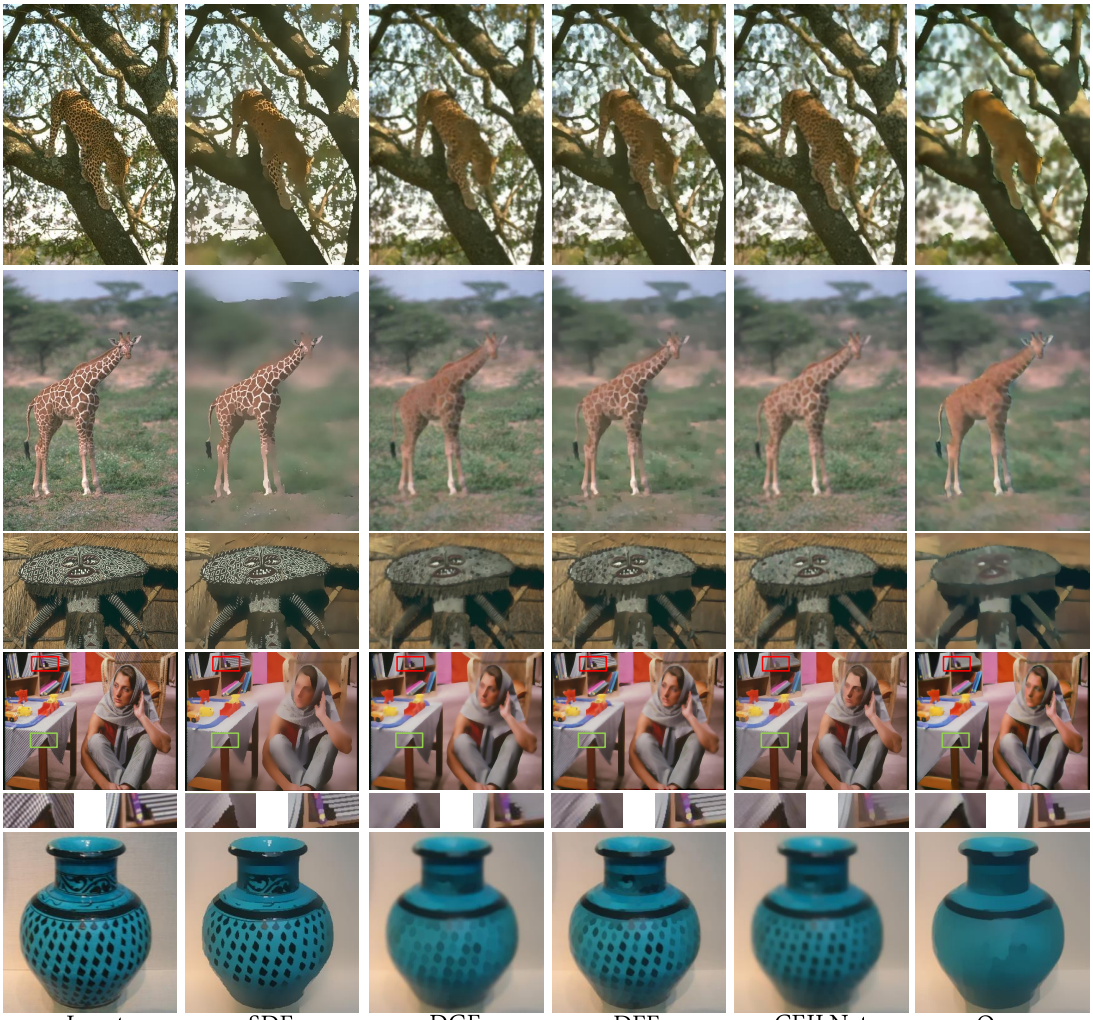



Figure 4.6: Smoothing results on natural images. The methods we compare are SDF [Ham et al., 2017], DGF [Lu et al., 2017], DFF [Chen et al., 2017a], and CEILNet [Fan et al., 2017b]. The first example shows the ability of weak structure preservation and enhancement in textured scenes. The next four examples present various texture types with different shapes, contrast, and distortion. Our filter performs consistently better in removing textures without degrading main structures.

### 4.4.3 Model Analysis & Ablation Studies

**Effect of each guidance.** To investigate the effect of each guidance, we train the filtering network with no guidance (baseline), only structure guidance, only texture guidance, and two guidance respectively. We report the average PSNR and SSIM of testing results in Table 4.3. Clearly, the use of two guidance produces better quantitative results. Also, fine-tuning further improves the filtering network by facilitating the synergy among three sub-networks. Additionally, we display two natural images in Fig. 4.7. Compared with the baseline without guidance, the result only with struc-

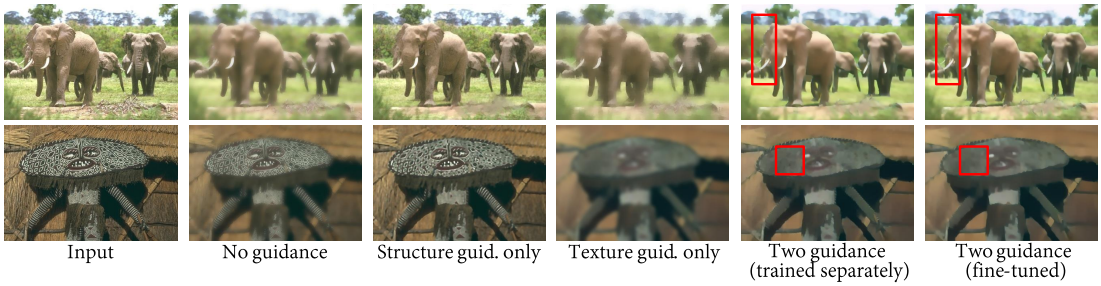



Figure 4.7: Image smoothing results with no guidance, only structure guidance, only texture guidance, and two guidance (trained separately, and fine-tuned). With only structure guidance, the main structures are retained as well as some strong textures. With only texture guidance, textures are mostly smoothed out but the structures are severely blurred. The use of two guidance leads to better structure preservation and texture removal. Fine-tuning the whole network can further improve the performance. The two images are from the BSDS dataset [Arbelaez et al., 2010].

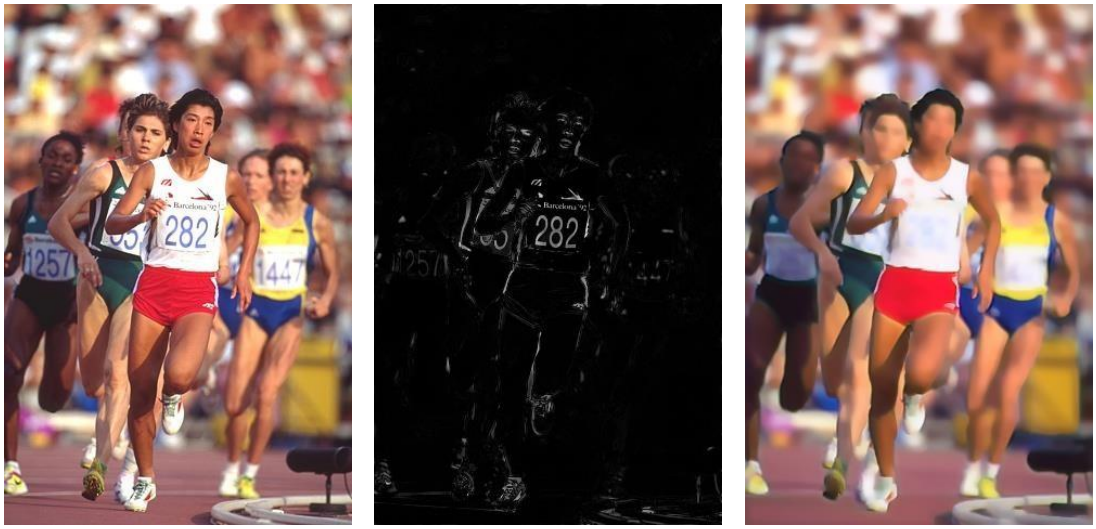
ture guidance retains more structures, as well as textures (this is mainly because HED [Xie and Tu, 2015] may also be negatively affected by strong textures). By contrast, the structures are severely blurred with only texture guidance, even though most textures are suppressed. Combining both structure guidance and texture guidance produces a better smoothing effect in both texture removal and structure preservation[2]. Fine-tuning further improves results (in the red rectangle of the first example, the structures are sharper; in the second example, the textures within the red region are further smoothed). Both quantitative and visual results demonstrate the effectiveness of simultaneously employing the two guidance. This conclusion is also consistent with the use of double guidance, *i.e.*, DGF [Lu et al., 2017], in Chapter 3.

**Challenging case.** We provide a challenging case in Fig. 4.8, where the eyes, nose, and the number of the runner are removed as textures. Nevertheless, they have important semantic meaning in the real world, but our texture prediction network cannot distinguish such a high-level semantic. This could motivate a future direction, *i.e.*, preserving semantically meaningful textures.

## 4.5 Conclusion

In this chapter, we have proposed an end-to-end texture and structure aware filtering network that is able to smooth images with both “texture-awareness" and “structure-awareness". “Texture-awareness" benefits from the proposed texture prediction network. To facilitate training, natural textures are blended with structure-only cartoon images with spatial and color variations. “Structure-awareness" is realized by semantic edge detection. Experimental results have demonstrated that the texture network can detect various types of textures effectively, showing good robustness to contrast,

[2]We provide more ablation results on synthetic images in Fig. 4.11 and natural images in Fig. 4.12.



(a) Input (b) Texture map (c) Smoothed output

Figure 4.8: Challenge case. The texture prediction network cannot distinguish semantically meaningful textures, *e.g.*, eyes, nose, and numbers. They are smoothed out in the output. The image is from the BSDS dataset [Arbelaez et al., 2010].

scales, and distribution of these textures. Due to the good discrimination of structures and textures, our filtering network outperforms both non-learning and learning filters on synthetic and natural images. Our future work will focus on incorporating more high-level semantic information into the image smoothing network.

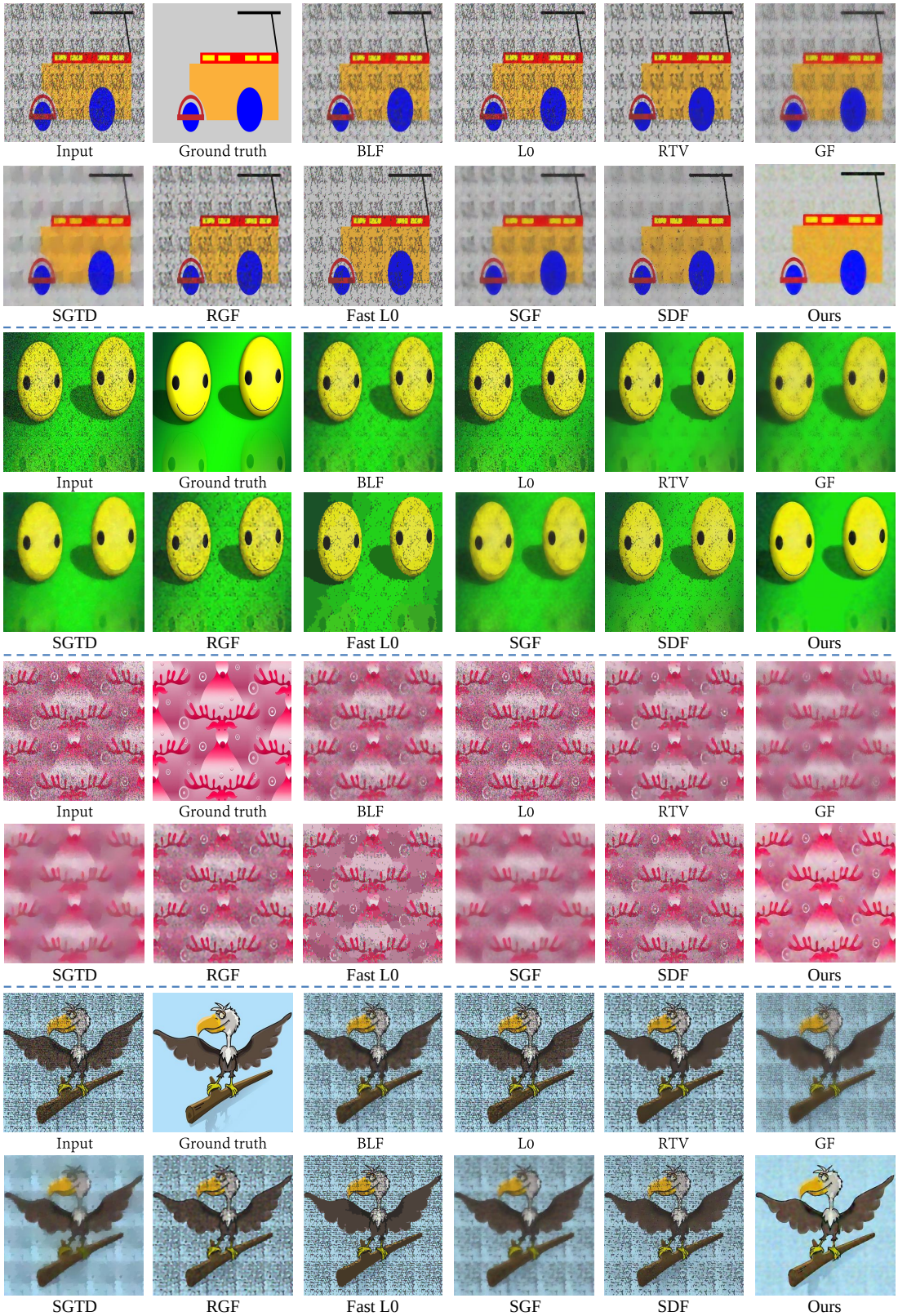



Figure 4.9: More image smoothing results with different methods on synthetic images.


Figure 4.10: More image smoothing results with different methods on natural images. These images are all from the BSDS dataset [Arbelaez et al., 2010].

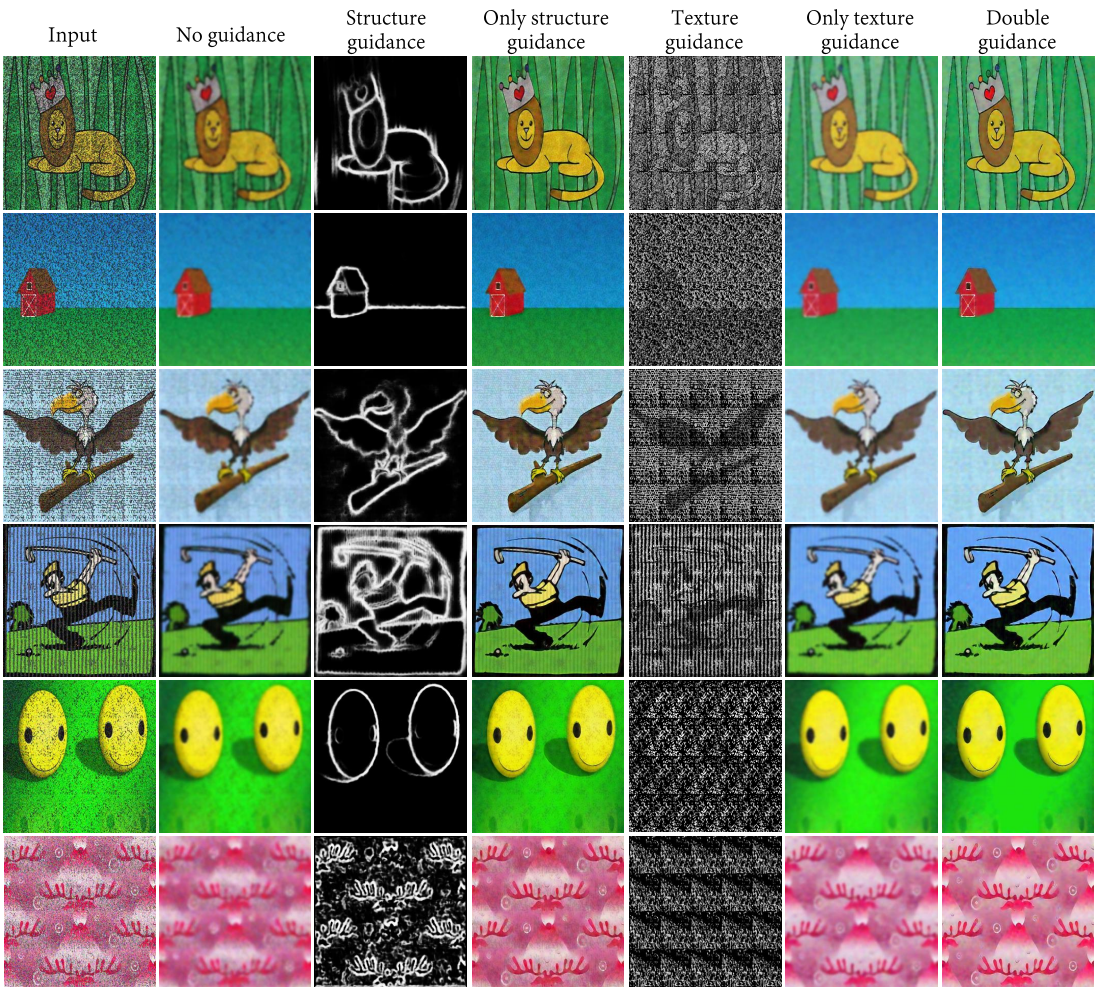



Figure 4.11: More ablation studies on synthetic images.

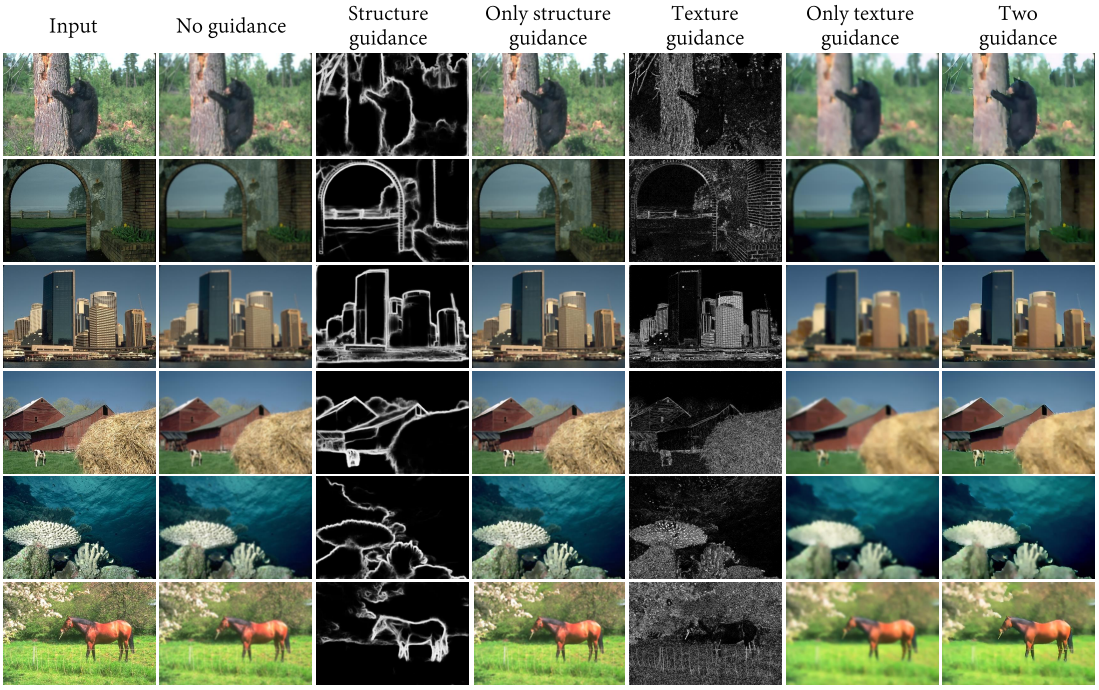



Figure 4.12: More ablation studies on natural images. These images are all from the BSDS dataset [Arbelaez et al., 2010].

Chapter 5

# Supervised Depth Completion via Auxiliary Image Reconstruction

In the last two chapters, we focused on the reduction of structure degradation in image smoothing by leveraging additional structure guidance and texture guidance. To further verify the effectiveness of using guidance in image enhancement, from the following two chapters, we will introduce a new enhancement task, *i.e.*, depth completion that recovers dense depth from sparse measurements. The main focus is on dealing with the structure degradation issue, *i.e.*, failing to produce semantically consistent boundaries or small/thin objects, in depth-only models that only take sparse depth as input. Our work continues the depth-only paradigm and aims to reduce structure degradation in both supervised (this chapter) and unsupervised (Chapter 6) settings. Our research handles a fundamental and challenging problem in depth completion, *i.e.*, how to improve performance without the image as an extra input. Besides, it is practical for some real-world applications where RGB images are not available at test time or images have degraded quality due to poor calibration between image and depth, bad weather, and nighttime.

In this chapter, we introduce a novel supervised depth completion model. The unique design is that it simultaneously outputs a reconstructed image and a dense depth map. Specifically, we formulate image reconstruction from sparse depth as an auxiliary task during training that is supervised by the unlabelled image. During testing, our system accepts sparse depth as the only input, *i.e.*, the image is not required. Our design enables the depth completion network to learn complementary image features that help to better understand object structures. The extra supervision incurred by image reconstruction is minimal, because no annotations other than the image are needed. We evaluate our method on the KITTI Depth Completion Benchmark [Uhrig et al., 2017] and show that depth completion can be significantly improved via auxiliary image reconstruction. Our model outperforms depth-only methods and is also suitable for indoor scenes like NYUv2 [Silberman et al., 2012].

In the remainder of this chapter, Section 5.1 introduces our motivation and summarizes primary contributions. Section 5.2 gives additional review on multi-task learning and its variant, auxiliary learning. Section 5.3 illustrates our method. We provide experimental results, model analysis, and ablation studies in Section 5.4.

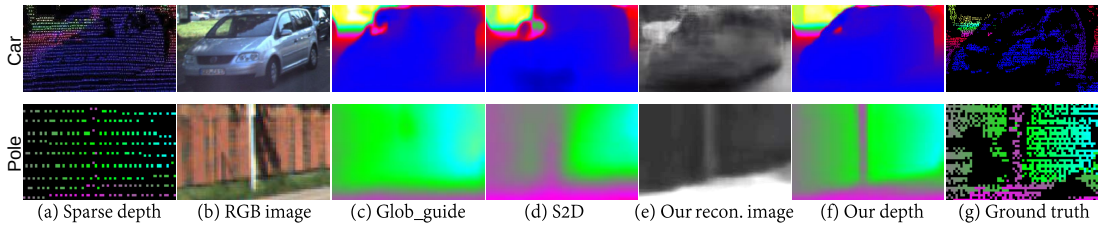



Figure 5.1: Depth completion from sparse depth. Only given (a) sparse depth as input without (b) the corresponding image, existing depth-only methods, like (c) Glob_guide [Van Gansbeke et al., 2019] and (d) S2D [Ma et al., 2019] cannot appropriately complete depth of objects with specific boundaries (*e.g.*, the car) and small/thin objects (*e.g.*, the pole), due to the lack of depth points and no images to provide structural cues. (e) Different from theirs, we recover these structural cues via image reconstruction directly from sparse depth. It helps (f) our depth completion recover more semantically consistent boundaries and small/thin objects more accurately. Our results are closer to (g) ground truth. All depth maps are colorized for better visualization.

Section 5.5 summarizes the chapter and proposes the future work.

## 5.1 Introduction

Depth-only depth completion models [Uhrig et al., 2017; Ma et al., 2019; Eldesokey et al., 2019] suffer from structure degradation when recovering dense depth only from the sparse input. Essentially, this issue results from the high input sparsity, which cannot provide sufficient information to localize object structures, as illustrated in Fig. 1.5 and Fig. 5.1. Existing works [Xu et al., 2019; Qiu et al., 2019; Cheng et al., 2018] take the image as an additional input to the network (named *multiple-input methods*), and employ early or late fusion (see Fig. 1.6) to incorporate image features to depth. Resorting to the image at both training and testing phases, however, may arise practical concerns due to the complicated process of aggregating features from two modalities [Eldesokey et al., 2019; Qiu et al., 2019] and highly-expensive depth-image calibration [Henry et al., 2012; Kerl et al., 2015].

The question arising from above is, can we continue the depth-only paradigm but incorporate more image features so as to provide richer structure information to overcome the shortcomings of this paradigm? To answer this, we start from an observation that, from sparse depth we can still roughly see some object structures according to their general shape and depth difference to the background, *e.g.*, car and pole examples in Fig. 5.1. This motivates us into thinking if some image structures can be recovered from sparse depth, we will be able to relax the need of taking the image as input.

Motivated by the above considerations, we propose a depth completion model that takes sparse depth as the only input and at the same time has the ability to learn from image features to provide structural cues. Specifically, we train the network to

output **a reconstructed image** and **a dense depth map** simultaneously, as illustrated in Fig. 1.6(a). We formulate image reconstruction from sparse depth as an auxiliary task during training that is **supervised by the unlabelled image**. During testing, no image is required. The unique design of our model allows the depth completion network to learn complementary image features that help to better understand object structures, and thus, produce more semantically consistent and accurate results than existing depth-only methods (see Fig. 5.1). Moreover, the extra supervision incurred by image reconstruction at the training stage is minimal, because no annotations other than the image are needed. Therefore, our method is practical in use. We evaluate our method on the KITTI Depth Completion Benchmark [Uhrig et al., 2017] and show that depth completion can be significantly improved via the auxiliary learning of image reconstruction.

In summary, we make the following major contributions:

- We propose a depth completion network that only takes sparse depth as input and outputs a reconstructed image and a dense depth map simultaneously. This practice largely overcomes the shortcomings of existing depth-only methods, *i.e.,* the lack of structural cues.
- By formulating image reconstruction as an auxiliary task during training, we do not need additional annotations other than the image. This is cheap and easy to implement. During testing, no image is required.
- We demonstrate that our approach significantly outperforms depth-only methods on the KITTI Depth Completion Benchmark and can be applied to indoor scenes.

## 5.2 Related Work

The related work on depth completion is introduced in Section 2.2. In this section, we add the brief review of multi-task learning and its variant, *i.e.*, auxiliary learning. The latter is directly related to our network training.

**Multi-task learning.** Multi-Task Learning (MTL) aims to improve performance by learning individual yet related tasks simultaneously [Argyriou et al., 2007]. Features are shared among these tasks to exploit common representations, while they can also be complementary to each other [Kendall et al., 2018]. This learning strategy has been successfully employed in semantic segmentation [Kendall et al., 2018; Pham et al., 2019], object detection [Lee et al., 2019b; Liang et al., 2019], single image depth estimation [Atapour-Abarghouei and Breckon, 2019; Zhang et al., 2019b], and so on.

**Auxiliary learning.** Recently, a variant of MTL, known as Auxiliary Learning (AL), is becoming popular. In this framework, a primary task is defined while all other tasks serve as auxiliary regularizers that enhance the primary one [Romera-Paredes et al., 2012]. AL has been proven to be effective in a number of computer vision tasks, *e.g.*, hand-written digit recognition [Zhang et al., 2014a], semantic segmentation [Liebel and Körner, 2018], face anti-spoofing [Liu et al., 2018b], visual

odometry [Valada et al., 2018]. We also employ it and focus on depth completion as the primary task. We expect the auxiliary task, *i.e.*, image reconstruction, to facilitate it with complementary image features that can help to better understand object structures. To the best of our knowledge, our work is the first one to introduce auxiliary learning to depth completion.

## 5.3 Methodology

In this section, we first give a general formulation to describe existing supervised depth completion models and contrast them with ours. We then illustrate the details of our method.

### 5.3.1 Depth Completion Models

Given a sparse depth map $\mathbf{x}$ where the empty locations are filled with zeros, a general depth completion model learns to recover dense depth $\tilde{\mathbf{x}}$ supervised by its ground truth $\mathbf{x}^*$.

**Depth-only model.** A depth-only model $D$ only takes sparse depth, $\mathbf{x}$, as input:

$$\tilde{\mathbf{x}} = D(\mathbf{x}; \theta_D), \tag{5.1}$$

where $\theta_D$ denotes the model parameters. The optimal model is parameterized by $\theta_D^*$, and obtained during training by minimizing the loss function $\mathcal{L}$, *i.e.*,

$$\theta_D^* = \arg\min_{\theta_D} \mathcal{L}(\tilde{\mathbf{x}}, \mathbf{x}^*). \tag{5.2}$$

**Multiple-input model.** A multiple-input model $T$ combines sparse depth $\mathbf{x}$ and the corresponding calibrated image $\mathbf{r}$ as input:

$$\tilde{\mathbf{x}} = T(\mathbf{x}, \mathbf{r}; \theta_T), \tag{5.3}$$

and the optimal model is

$$\theta_T^* = \arg\min_{\theta_T} \mathcal{L}(\tilde{\mathbf{x}}, \mathbf{x}^*). \tag{5.4}$$

**Our model.** As illustrated in Fig. 5.2, our model $G$ takes sparse depth $\mathbf{x}$ as the only input, and outputs dense depth $\tilde{\mathbf{x}}$ and a reconstructed image $\tilde{\mathbf{r}}$ simultaneously:

$$\tilde{\mathbf{x}}, \tilde{\mathbf{r}} = G(\mathbf{x}; \theta_G) \Rightarrow \begin{cases} \tilde{\mathbf{x}} = G_{dpt}(\mathcal{F}(\mathbf{x}; \theta_{\mathcal{F}}); \theta_{dpt}, \theta_{shr}) \\ \tilde{\mathbf{r}} = G_{img}(\mathcal{F}(\mathbf{x}; \theta_{\mathcal{F}}); \theta_{img}, \theta_{shr}) \end{cases}, \tag{5.5}$$

where $\mathcal{F}$ parameterized by $\theta_{\mathcal{F}}$ extracts features from the input, $\theta_{dpt}$ and $\theta_{img}$ are parameters for the depth completion module $G_{dpt}$ and image reconstruction module $G_{img}$ respectively, and $\theta_{shr}$ represents feature sharing between the two modules. During training, the parameters of the joint model, $\theta_G = (\theta_{\mathcal{F}}, \theta_{dpt}, \theta_{img}, \theta_{shr})$, are

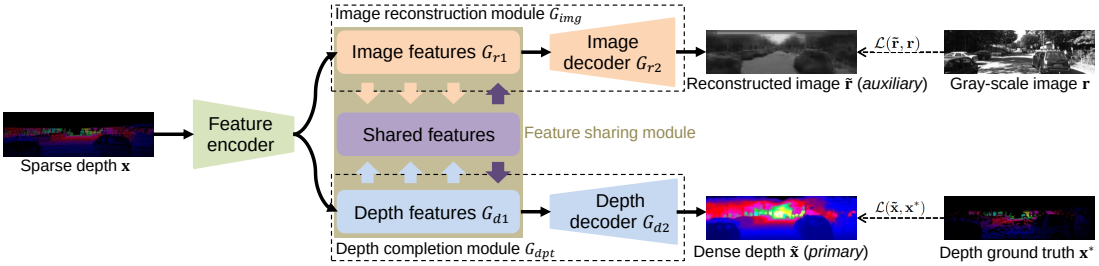



Figure 5.2: Network architecture for *training* our model. It contains: (1) the feature encoder - extracting initial features from the sparse input; (2) the depth completion module - specializing depth features and producing dense depth; (3) the image reconstruction module - specializing image features and reconstructing the image from sparse depth; and (4) the feature sharing module - aggregating features from depth and image modules and transferring them to each module. Depth completion is the primary task, while image reconstruction is an auxiliary task and supervised by the gray-scale image.

optimized such that

$$\theta_G^* = \arg\min_{\theta_G}(w_{dpt} \cdot \mathcal{L}(\tilde{\mathbf{x}}, \mathbf{x}^*) + w_{img} \cdot \mathcal{L}(\tilde{\mathbf{r}}, \mathbf{r})), \tag{5.6}$$

where $w_{dpt}$ and $w_{img}$ are weighting factors of the two tasks. This is a typical multi-task learning framework [Argyriou et al., 2007], where the network jointly learns to recover dense depth and reconstruct the image *directly* from the sparse input. More specifically, we treat depth completion as the primary task, and image reconstruction as an auxiliary task, which is known as *auxiliary learning* [Romera-Paredes et al., 2012]. The purpose is to transfer useful knowledge from the auxiliary task to the primary one to enhance the feature learning of the latter [Dai et al., 2007]. In our case, by enforcing feature correlations via sharing, we expect the depth completion network to learn more complementary image features to provide structural cues for understanding object structures. Note that the auxiliary image reconstruction is supervised by unlabelled camera images, which are cheaper to acquire than manually-labelled data. In the following, we illustrate the network architecture, loss functions, and how image reconstruction facilitates depth completion.

During testing, we only focus on the primary depth completion and no image is required, *i.e.*,

$$\tilde{\mathbf{x}} = G_{dpt}(\mathcal{F}(\mathbf{x}; \theta_{\mathcal{F}}^*); \theta_{dpt}^*, \theta_{shr}^*). \tag{5.7}$$

### 5.3.2 Network Architecture

The overall network architecture for *training* our model is based on Eq. 5.5 and Eq. 5.6, and illustrated in Fig. 5.2.

**Feature encoder $\mathcal{F}$.** We extract multi-scale features from the input by convolving it with different kernel sizes. This is inspired by the Inception architecture [Szegedy

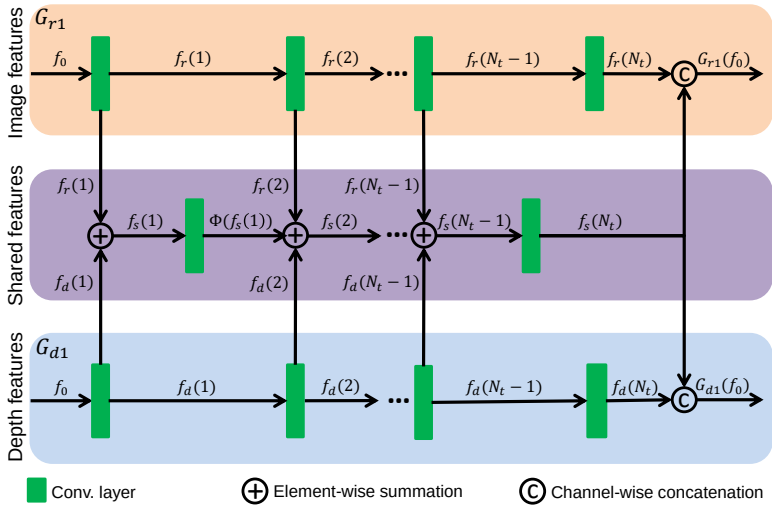



Figure 5.3: Structure of the feature sharing module. It aggregates depth and image features by element-wise summation, followed by convolutions in each layer. The depth and image feature modules output the concatenation of their last layer features and the shared features.

et al., 2015], but with $3 \times 3$, $5 \times 5$, $7 \times 7$, $9 \times 9$ kernels instead. In the last layer, all the feature maps are with $1/16$ resolution to the input and concatenated in a channel-wise manner. We denote the output of this encoder, representing initial features from the sparse input, as $f_0 = \mathcal{F}(\mathbf{x})$.

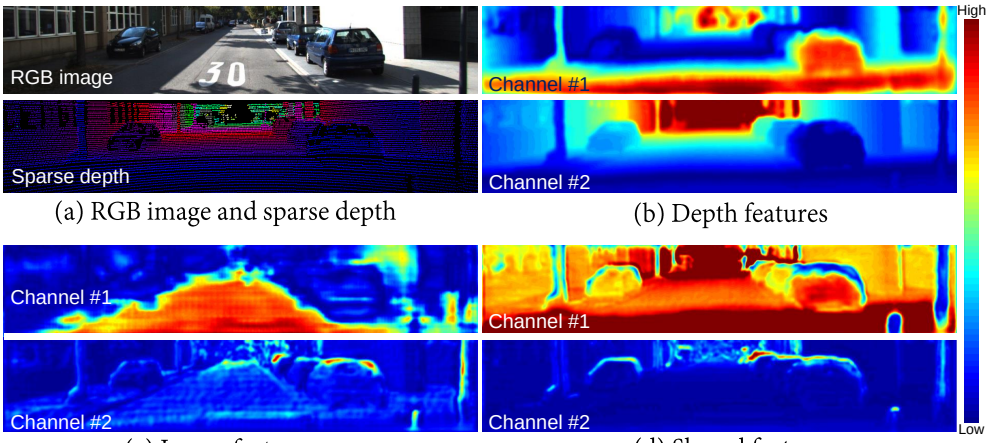
**Depth completion module $G_{dpt}$.** It is composed of a depth feature extractor $G_{d1}$ and depth decoder $G_{d2}$. $G_{d1}$ focuses on learning depth-specific features and gradually upsamples $f_0$ with transpose convolutions ($1/16 \rightarrow 1/8 \rightarrow 1/4 \rightarrow 1/2$). The intermediate features in $G_{d1}$ are also transferred to the feature sharing module (see Fig. 5.3). Its output, $G_{d1}(f_0)$, containing both depth and shared features, is fed into $G_{d2}$ to produce dense depth.

**Image reconstruction module $G_{img}$.** The underlying architecture of the image reconstruction module is identical to the depth completion module, where $G_{r1}$ specializes and transfers image features. The image decoder, $G_{r2}$, outputs the reconstructed image based on image-specific and shared features.

**Feature sharing module.** This module aggregates features from depth and image feature modules via element-wise summation followed by convolutions in each layer, as illustrated in Fig. 5.3. Suppose there are $N_t$ layers in each module, and we denote the feature maps in *n-th* convolutional layer in $G_{d1}$, $G_{r1}$, and the sharing module as $f_d(n)$, $f_r(n)$, and $f_s(n)$ respectively. We use $\Phi(\cdot)$ to represent the general convolutional operator. In the first layer, *i.e.*, $n = 1$,

$$\begin{cases} f_r(1) = \Phi(f_0) \\ f_d(1) = \Phi(f_0) \\ f_s(1) = f_r(1) \oplus f_d(1) \end{cases}, \tag{5.8}$$

where $\oplus$ is element-wise summation. In subsequent layers before the last layer, *i.e.*,




(a) RGB image and sparse depth (b) Depth features

(c) Image features (d) Shared features

Figure 5.4: Feature visualization. (a) The RGB image is used for reference and sparse depth is the only input. (b) Depth features emphasize more on objects that are visible in both near and far regions of the depth map. (c) Image features highlight global visual structures as well as some details that are not reflected in depth. (d) Shared features take advantage of both depth and image features, and cover most objects (upper) as well as some details like their boundaries (bottom).

$1 < n < N_t$,

$$\begin{cases} f_r(n) = \Phi(f_r(n-1)) \\ f_d(n) = \Phi(f_d(n-1)) \\ f_s(n) = f_r(n) \oplus f_d(n) \oplus \Phi(f_s(n-1)) \end{cases} . \tag{5.9}$$

In the last layer where $n = N_t$, only convolutions are performed,

$$\begin{cases} f_r(N_t) = \Phi(f_r(N_t-1)) \\ f_d(N_t) = \Phi(f_d(N_t-1)) \\ f_s(N_t) = \Phi(f_s(N_t-1)) \end{cases} . \tag{5.10}$$

The final output of both $G_{d1}$ and $G_{r1}$ is the channel-wise concatenation of their corresponding feature maps and the shared features, *i.e.*,

$$\begin{cases} G_{d1}(f_0) = Cat(f_d(N_t), f_s(N_t)) \\ G_{r1}(f_0) = Cat(f_r(N_t), f_s(N_t)) \end{cases} . \tag{5.11}$$

The two concatenated features are further fed into depth and image decoders to produce dense depth $\tilde{\mathbf{x}}$ and the reconstructed image $\tilde{\mathbf{r}}$, *i.e.*,

$$\begin{cases} \tilde{\mathbf{x}} = G_{dpt}(f_0) = G_{d2}(G_{d1}(\mathcal{F}(\mathbf{x}))) \\ \tilde{\mathbf{r}} = G_{img}(f_0) = G_{r2}(G_{r1}(\mathcal{F}(\mathbf{x}))) \end{cases} . \tag{5.12}$$

**Loss functions.** To train the network, we first define the $\ell_2$ loss for depth com-

pletion (primary task):

$$\ell_{dpt} = \frac{1}{N_1} \|\Psi \odot (\tilde{\mathbf{x}} - \mathbf{x}^*)\|_2^2, \tag{5.13}$$

where $N_1$ is the number of pixels that have depth values in ground truth $\mathbf{x}^*$, $\Psi$ is a binary mask of $\mathbf{x}^*$ where 1 means the input pixel has a depth value and 0 for none, and $\odot$ is the element-wise multiplier. We use the gray-scale image, $\mathbf{r}$, to supervise auxiliary image reconstruction. The $\ell_2$ loss function is:

$$\ell_{img} = \frac{1}{N_2} \|\tilde{\mathbf{r}} - \mathbf{r}\|_2^2, \tag{5.14}$$

where $N_2$ is the number of pixels in the image. Hence, the total loss for the entire network is:

$$\ell_{total} = w_{dpt} \cdot \ell_{dpt} + w_{img} \cdot \ell_{img}. \tag{5.15}$$

Eq. 5.15 indicates that $\ell_{img}$ serves as a regularizer during training to facilitate parameter learning of depth completion and thus improves its overall performance.

### 5.3.3 Discussion

To further investigate the learning ability of our network, we select and visualize two representative feature maps from the first and second channels in the last layers of depth features, image features, and shared features respectively, *i.e.*, $f_d(N_t)$, $f_r(N_t)$, and $f_s(N_t)$. Depth features shown in Fig. 5.4(b) indicate that they emphasize more on visible objects in both near and far regions of the depth map, *e.g.*, cars and poles. However, due to the sparsity of depth points and lack of image information, these features only partially reflect the real shape of these objects.

Image features in Fig. 5.4(c), by contrast, highlight the global structure, *e.g.*, the road, and some details that are not reflected in depth, *e.g.*, the missing parts around car boundaries and poles. These features are beneficial for better distinguishing object boundaries and recovering the full structure of small/thin objects. Therefore, image features are complementary to depth features. After aggregating these features via the sharing module, the shared features shown in Fig. 5.4(d) take advantage of both depth and image features, *i.e.*, covering most objects as well as some details like their boundaries. In summary, the auxiliary learning of image reconstruction enables the depth completion network to learn useful and complementary image features via sharing, and thus obtains more structural cues for better completion. This can be achieved even without the image as input.

## 5.4 Experiments

In the following, we show the effectiveness of our method through extensive experiments. This includes quantitative and visual comparison with state-of-the-art approaches, ablation studies on several factors that affect completion performance, and the application to indoor scenes.

| Method | RMSE ↓ | MAE ↓ | iRMSE ↓ | iMAE ↓ |
|---|---|---|---|---|
| SparseConvs [Uhrig et al., 2017] | 1601.33 | 481.27 | 4.94 | 1.78 |
| ADNN [Chodosh et al., 2018] | 1325.37 | 439.48 | 59.39 | 3.19 |
| Spade-sD [Jaritz et al., 2018] | 1035.29 | 248.32 | **2.60** | **0.98** |
| NConv-CNN (d) [Eldesokey et al., 2019] | 1268.22 | 360.28 | 4.67 | 1.52 |
| S2D (d) [Ma et al., 2019] | 954.36 | 288.64 | 3.21 | 1.35 |
| Glob_guide [Van Gansbeke et al., 2019] | 922.93 | 249.11 | 2.80 | 1.07 |
| Ours ($\ell_2$ loss) | **901.43** | 292.36 | 4.92 | 1.35 |
| Ours ($\ell_1$ loss) | 915.86 | **231.37** | 3.19 | 1.23 |
| DeepLiDAR [Qiu et al., 2019] | 758.38 | 226.50 | 2.56 | 1.15 |
| PwP [Xu et al., 2019] | 777.05 | 235.17 | 2.42 | 1.13 |
| S2D (gd) [Ma et al., 2019] | 814.37 | 249.95 | 2.80 | 1.21 |
| NConv-CNN (gd) [Eldesokey et al., 2019] | 829.98 | 233.26 | 2.60 | 1.03 |
| CSPN [Cheng et al., 2018] | 1019.64 | 279.46 | 2.93 | 1.15 |

Table 5.1: Quantitative comparison with state-of-the-art methods on the KITTI *test* set. The best results are marked with **bold** among methods that do not use any images during testing (gray region). ↓ means smaller is better.

### 5.4.1 Implementation Details

**Dataset.** The KITTI Depth Completion Benchmark [Uhrig et al., 2017] contains raw, sparse depth maps collected by LiDAR which are further separated into 85,898 frames for training, 1,000 for validation, and 1,000 for testing. Each depth map has the corresponding RGB image, and we convert the RGB image to gray-scale to supervise image reconstruction only at the training stage (we empirically find that using the gray image for supervision generates slightly better results, which is consistent with the observation in [Ma et al., 2018]). The KITTI ground truth is generated by accumulating multiple LiDAR frames, and removing outliers by semi-global matching [Uhrig et al., 2017]. Hence, depth ground truth is semi-dense (depth completion becomes harder in this case because semi-dense ground truth cannot completely reflect the depth of some object boundaries and small objects). Test samples have no ground truth available, and the results are evaluated on the benchmark server.

**Training configuration.** The network is implemented in PyTorch [Paszke et al., 2017]. During training, the input is cropped from the bottom to $352 \times 1216$. We train the network on two NVIDIA 1080 Titan GPUs with a batch size of 16. The loss function is defined in Eq. 5.15, where $w_{dpt} = 1$ and $w_{img} = 10^{-4}$. We use the Adam optimizer [Kingma and Ba, 2014], and the initial learning rate is $10^{-3}$ and decayed by half every five epochs.

**Evaluation metrics.** Following the benchmark [Uhrig et al., 2017], we use four evaluation metrics: (1) rooted mean squared error (RMSE): $\sqrt{\frac{1}{N_1}\sum_{i=1}^{N_1}(x_i^* - \tilde{x}_i)^2}$; (2) mean absolute error (MAE): $\frac{1}{N_1}\sum_{i=1}^{N_1}|x_i^* - \tilde{x}_i|$; (3) inverse rooted mean squared error (iRMSE): $\sqrt{\frac{1}{N_1}\sum_{i=1}^{N_1}(\frac{1}{x_i^*} - \frac{1}{\tilde{x}_i})^2}$; (4) inverse mean absolute error (iMAE): $\frac{1}{N_1}\sum_{i=1}^{N_1}\left|\frac{1}{x_i^*} - \frac{1}{\tilde{x}_i}\right|$. Here $x^*$, $\tilde{x}$, and $N_1$ represent ground truth depth, predicted depth, and the number

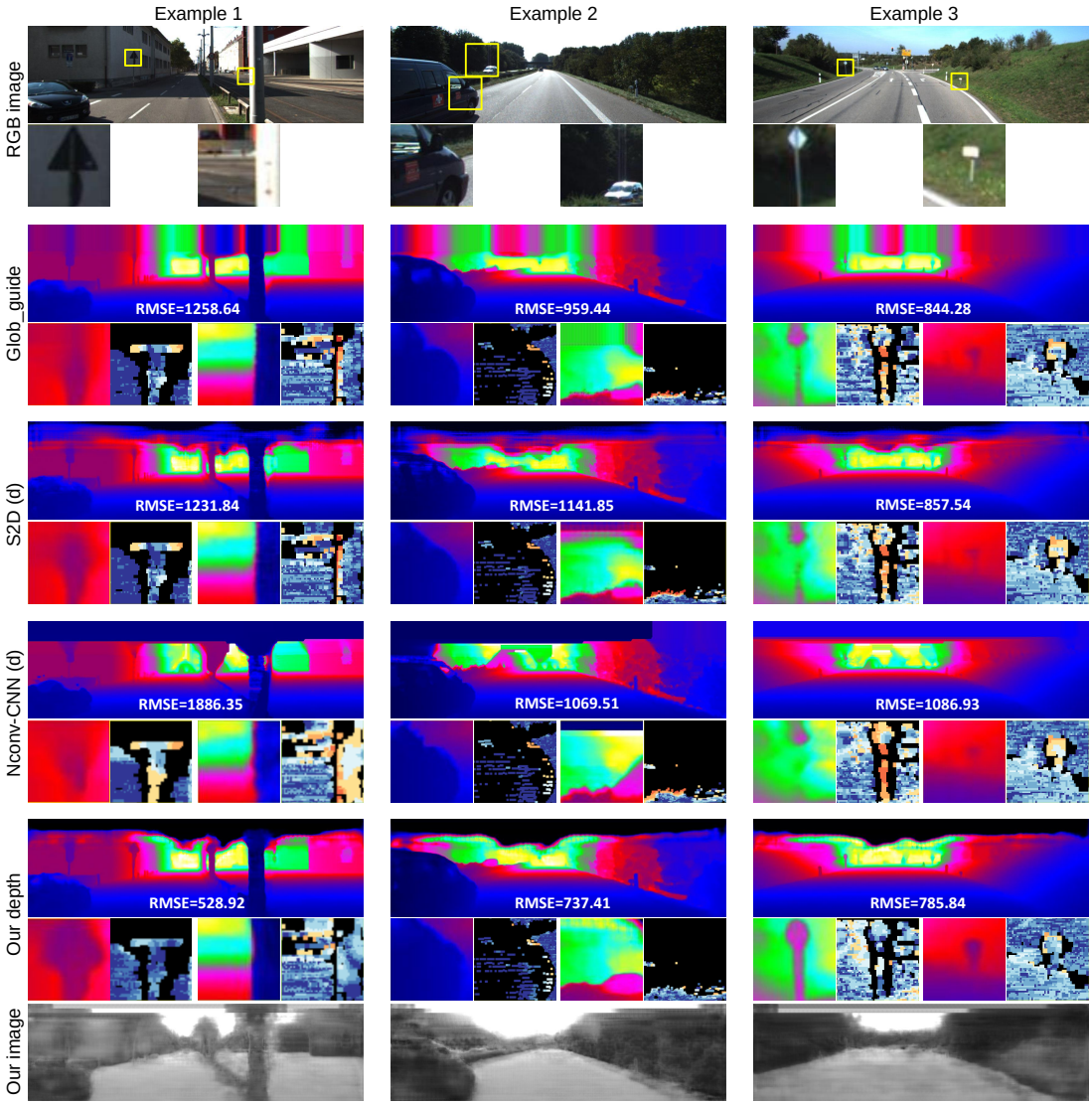
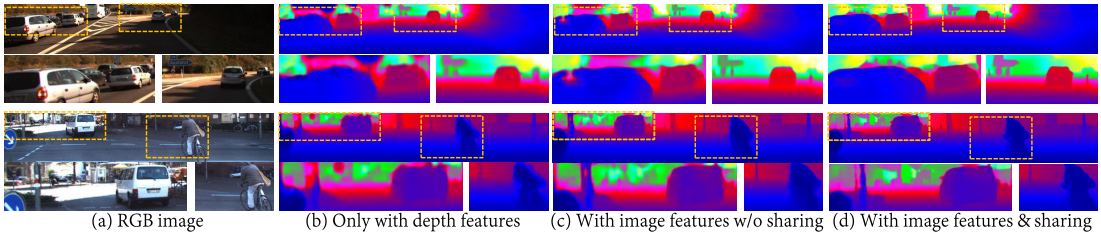



Figure 5.5: Visual comparison with state-of-the-art depth-only methods on the KITTI *test* set. The methods we compare include Glob_guide [Van Gansbeke et al., 2019], S2D (d) [Ma et al., 2019], and NConv-CNN (d) [Eldesokey et al., 2019]. Our model can produce more accurate depth completion results in small/thin objects, boundaries, and distant regions. To the right of each close-up is the error map, where small errors are displayed in blue and large errors in red. Black regions mean the ground truth labels are not used for evaluation. The contrast of our reconstructed images has been enhanced for better visualization.



(a) RGB image (b) Only with depth features (c) With image features w/o sharing (d) With image features & sharing

Figure 5.6: Visual comparison of depth completion results after incorporating image reconstruction and feature sharing. (a) RGB images for reference. (b) Only with depth features cannot recover the full structure of objects. (c) With image features but without sharing, the results are slightly improved. (d) With shared features, the model performs better in recovering consistent object structures and small/thin objects.

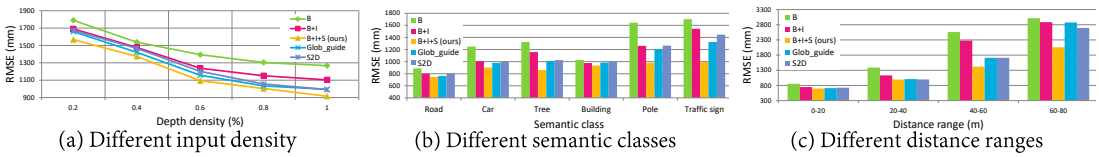
| | RMSE ↓ | MAE ↓ |
|---|---|---|
| B | 1267.01 | 322.32 |
| B + I | 1103.85 | 301.64 |
| B + I + S (ours) | **914.65** | **297.38** |

Table 5.2: Ablation study on the KITTI *validation* set. "B", "I", and "S" represent the baseline only with depth features, image features, and feature sharing respectively. The best results are marked with **bold**. ↓ means smaller is better.

of valid pixels in ground truth depth respectively. RMSE and MAE are measured by *mm*, and iRMSE and iMAE are measured by *1/km*. RMSE calculates depth completion errors directly and penalizes more on undesirable larger errors. Differently, MAE treats all the errors equally. Hence, we consider RMSE to be the more important metric, which is consistent with the benchmark where RMSE is used for ranking.

### 5.4.2 Comparison with Existing Methods

**Quantitative comparison.** In Table 5.1, we report quantitative results of our method as well as the state-of-the-art approaches on the KITTI test set. Compared with depth-only methods (highlighted in gray), our model trained with the $\ell_2$ loss achieves the best **RMSE = 901.43**, ranking first among them and surpassing the second place by 21.50 (2.33%). Our MAE and iMAE are both comparable to others. However, our iRMSE is less competitive. The underlying reason is iRMSE measures the accuracy of inverse depth, in which case depth points in closer regions with relatively smaller errors are more dominant. By contrast, we use the $\ell_2$ loss for depth to penalize larger errors. There thus exists a trade-off in balancing large and small errors with this metric. We consider that iRMSE is less reliable than RMSE in reflecting the model accuracy mainly because iRMSE is not a direct metric to measure depth errors. We refer the reader to Fig. 5.7(c) where our model performs competitively against the state-of-the-art methods in close regions, *e.g.*, 0-40m. iMAE has the same issue.




(a) Different input density (b) Different semantic classes (c) Different distance ranges

Figure 5.7: Quantitative comparison with the baseline and state-of-the-art methods Glob_guide [Van Gansbeke et al., 2019], S2D [Ma et al., 2019] in three cases on the KITTI *validation* set. "B", "I", and "S" represent baseline only with depth features, image features, and feature sharing respectively. Our model performs consistently better in all cases.

Consequently, we still consider RMSE as the primary metric.

In fact, several studies have observed that training with different loss functions may yield different results [Chen et al., 2019; Eldesokey et al., 2019]. For example, Spade-sD [Jaritz et al., 2018] achieves the best iRMSE and iMAE because it is directly trained on inverse depth. To further validate our method, we re-train the model with the $\ell_1$ loss with the same network setting in Section 5.4.1. Unsurprisingly, using the $\ell_1$ loss yields a smaller MAE (best among depth-only methods) but slightly larger RMSE (it still ranks first among depth-only methods). Since we mainly focus on RMSE, in the following, our default model refers to the one trained with the $\ell_2$ loss unless otherwise specified.

Our model is also comparable to multiple-input methods, *e.g.*, it surpasses CSPN [Cheng et al., 2018] in terms of RMSE, and outperforms PwP [Xu et al., 2019], S2D (gd) [Ma et al., 2019], NConv-CNN-L2 (gd) [Eldesokey et al., 2019], and CSPN [Cheng et al., 2018] in MAE if trained with the $\ell_1$ loss. In summary, our approach generally lies in between depth-only and multiple-input methods, showing competitive performance even without using the image as input.

**Visual comparison.** We present qualitative results in Fig. 5.5 and compare with three state-of-the-art depth-only methods, *i.e.*, Glob_guide [Van Gansbeke et al., 2019], S2D (d) [Ma et al., 2019], and Nconv-CNN (d) [Eldesokey et al., 2019]. For each example, we also provide the RMSE and close-ups (left) with corresponding error maps (right). Overall, our model is able to produce more accurate depth completion results for small/thin objects, boundaries, and distant regions. Specifically, our method recovers the depth of narrow poles in Example 1 and 3 more appropriately in preserving their general structures. Besides, our completion results also have smaller errors along boundaries of the tree and car, as well as the distant regions, *e.g.*, the right close-up in Example 2 where the white car and its surroundings are relatively far away.

Moreover, our RMSE in these three examples is significantly better than others. The good performance is mainly owing to image reconstruction as an auxiliary task[1], because it enables our depth completion network to acquire more image features and

[1]These reconstructed images displayed in Fig. 5.5 are less comparable to the original images from appearance. However, for image reconstruction, we only care about the object structures it can reveal, rather than the specific intensity.

| | RMSE ↓ | REL ↓ | $\delta_{1.25}$ ↑ | $\delta_{1.25^2}$ ↑ | $\delta_{1.25^3}$ ↑ |
|---|---|---|---|---|---|
| Bilateral [Silberman et al., 2012] | 0.479 | 0.084 | 92.4 | 97.6 | 98.9 |
| TVG [Ferstl et al., 2013] | 0.635 | 0.123 | 81.9 | 93.0 | 96.8 |
| Zhang *et al*. [Zhang and Funkhouser, 2018] | 0.228 | 0.042 | 97.1 | 99.3 | 99.7 |
| Ma *et al*. [Ma and Karaman, 2018] | 0.204 | 0.043 | 97.8 | 99.6 | 99.9 |
| Nconv-CNN [Eldesokey et al., 2019] | 0.129 | 0.018 | 99.0 | 99.8 | **100** |
| CSPN [Cheng et al., 2018] | 0.117 | **0.016** | 99.2 | **99.9** | **100** |
| DeepLiDAR [Qiu et al., 2019] | **0.115** | 0.022 | **99.3** | **99.9** | **100** |
| Ours | 0.125 | 0.030 | 99.1 | 99.8 | **100** |

Table 5.3: Quantitative comparison on the NYUv2 dataset. Note that ours is the only one that does not use the image during testing, while others take the image as an additional input at both training and testing stages. The best results are marked with **bold**. ↓ means smaller is better, and ↑ means larger is better.

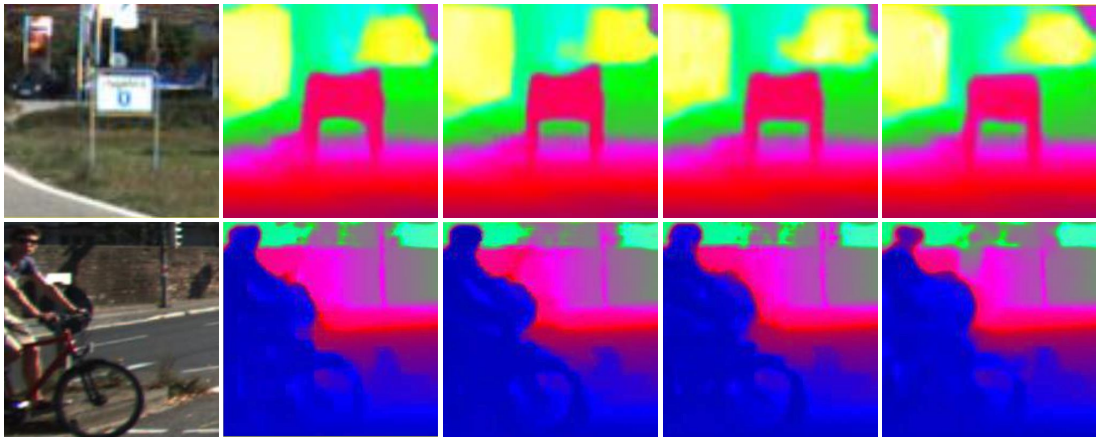
understand object structures better. Besides, since the image is truly dense, it can also overcome the shortcoming of semi-dense ground truth in reflecting the full structure of objects. Therefore, our performance is largely improved over depth-only methods. More visual results can be found in Fig. 5.9.

### 5.4.3 Model Analysis & Ablation studies

**Impact of image reconstruction.** Our proposed auxiliary image reconstruction can largely facilitate depth completion. To justify this, we set the baseline $B$ as the combination of the feature encoder and depth completion module. Based on it, $B + I$ denotes the incorporation of the image reconstruction module but without feature sharing, while $B + I + S$ is our ultimate model with shared features. The quantitative comparison in terms of RMSE and MAE is reported in Table 5.2. With only image reconstruction as an additional task but no shared features, depth completion performance is slightly boosted. This is mainly because more parameters are introduced but the image features are not sufficiently transferred to the depth completion network. Feature sharing between depth and image modules enables the depth completion network to better take advantage of image features, and thus the overall performance is further improved. Fig. 5.6 shows the qualitative comparison, where after feature sharing, the model performs better in recovering consistent object structures and small/thin objects.

**Robustness to input density.** We randomly drop depth points in the sparse input with different ratios, and compare RMSE with the baseline and other two state-of-the-art methods Glob_guide [Van Gansbeke et al., 2019] and S2D [Ma et al., 2019] in Fig. 5.7(a). Our model performs consistently better than others, indicating its robustness to input sparsity.

**Comparison in different semantic classes.** To validate that our model is able to acquire semantically meaningful image features and use them to facilitate depth completion, we compare results within different semantic classes. Specifically, we fine-



(a) RGB image (b) DeepLiDAR (c) PwP (d) S2D (gd) (e) Ours

Figure 5.8: Visual comparison with state-of-the-art multiple-input methods on the KITTI *test* set. The methods we compare are DeepLiDAR [Qiu et al., 2019], PwP [Xu et al., 2019] and S2D (gd) [Ma et al., 2019].

tune the off-the-shelf PSPNet [Zhao et al., 2017] pre-trained on Cityscapes [Cordts et al., 2016] with 400 labelled images from the KITTI Semantic Segmentation Benchmark [Alhaija et al., 2018]. We use this model to generate semantic masks for the KITTI validation set. We calculate the RMSE of depth completion in six representative classes, *i.e.*, Road, Car, Tree, Building, Pole, and Traffic sign, as shown in Fig. 5.7(b). The performance in Road and Building classes of different methods is similar, mainly because these large and flat regions have more depth points in the input and thus are easier to complete. Our model performs significantly better than others in Car, Tree, Pole, and Traffic sign classes, which tend to have more specific boundaries and smaller structures. The good performance largely benefits from the effective understanding and incorporation of object structures with auxiliary image reconstruction and feature sharing.

**Results in different distance ranges.** Next, we compare completion results in different distance ranges. As illustrated in Fig. 5.7(c), our model performs slightly better in near regions (0-40m) but significantly better in distant regions (40-80m). This is mainly owing to (1) the use of the $\ell_2$ loss which penalizes more on larger errors that mostly exist in distant regions, and (2) image features can reflect the global structure like the road (see Fig. 5.4(c)) which facilitates our model with a better discrimination in near and distant regions. Besides, the results in the nearest regions, *i.e.*, 0-20m, are competitive to others, which is not properly reflected by iRMSE and iMAE.

**Application to indoor scenes.** We study the applicability of our model in indoor scenes, *i.e.*, NYUv2 [Silberman et al., 2012]. Following [Ma and Karaman, 2018], we only retain 500 points in each depth map, the same for other methods we compare. We re-train our network *from scratch* with this new dataset (nearly 50K images from 249 scenes for training, and 654 for testing). The evaluation metrics are RMSE, REL (mean absolute relative error), and the percentage of completed depth with both the

relative error and its inverse under a threshold $t$, *i.e.*, $t = 1.25, 1.25^2, 1.25^3$. Quantitative results are reported in Table 5.3. Note that all the methods for comparison take the RGB image as an additional input. Our model outperforms non-learning based Bilateral [Silberman et al., 2012] and TVG [Ferstl et al., 2013], and deep learning methods Zhang *et al*. [Zhang and Funkhouser, 2018], Ma *et al*. [Ma and Karaman, 2018] and NConv-CNN [Eldesokey et al., 2019] in terms of RMSE. Our performance is also comparable to CSPN [Cheng et al., 2018] and DeepLiDAR [Qiu et al., 2019]. In summary, our model can also be applied to other scenes, and thus is a generic approach for depth completion.

**Visual comparison with state-of-the-art multiple-input methods.** In Fig. 5.8, we display two examples compared with state-of-the-art multiple-input methods, *i.e.*, DeepLiDAR [Qiu et al., 2019], PwP [Xu et al., 2019], and S2D (gd) [Ma et al., 2019]. In the first example, we achieve competitive performance in recovering the depth of the traffic board, *i.e.*, our model produces even smoother boundary at the top. However, in the second example, our results are less comparable especially on the bicycle wheel and the human head. This is because these regions tend to have more complex structures and illumination changes, and the corresponding image reconstruction becomes more difficult. To address this issue, semantic segmentation results, *e.g.*, edges or labels, can be used to provide more specific structure information of objects. We will explore this in the future work.

## 5.5 Conclusion

In this chapter, we have proposed a depth completion model that takes sparse depth as the only input and outputs dense depth and a reconstructed image simultaneously. The auxiliary learning of image reconstruction from sparse depth during training enables the depth completion network to acquire more complementary image features for understanding object structures. On the KITTI Depth Completion Benchmark [Uhrig et al., 2017], our model has achieved competitive performance in producing more consistent boundaries and recovering the depth of small/thin objects more appropriately. It largely overcomes the shortcomings of existing depth-only approaches due to the lack of structural cues from images. Our model can also be applied to indoor scenes. Potential future work can be recovering other useful information directly from sparse depth if ground truth is available, *e.g.*, semantic labels, surface normal, to facilitate depth completion.

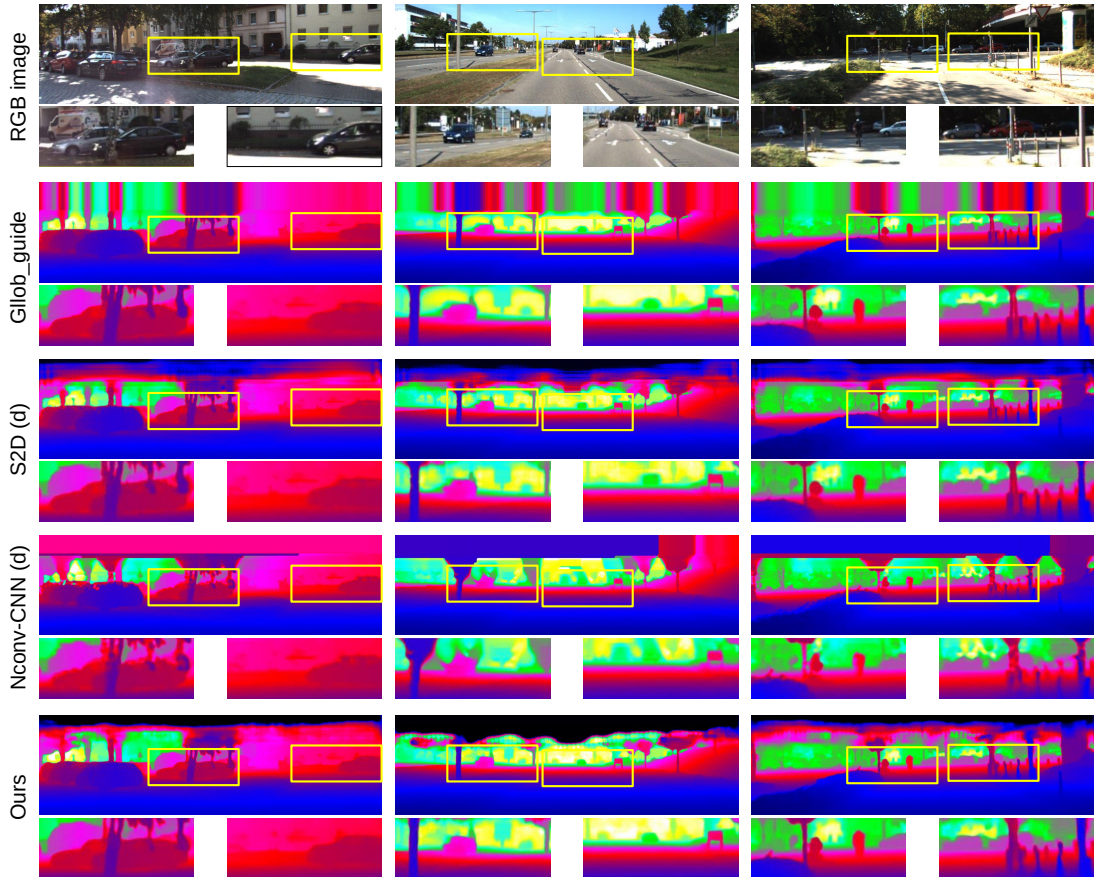



Figure 5.9: More visual comparison with state-of-the-art depth-only methods. The methods we compare are Glob_guide [Van Gansbeke et al., 2019], S2D (d) [Ma et al., 2019], and Nconv-CNN (d) [Eldesokey et al., 2019]. These results are obtained from the KITTI benchmark. Our method can produce more semantically-consistent depth values on boundaries and small/thin objects.

Chapter 6

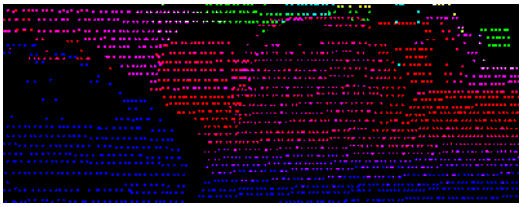
# Unsupervised Depth Completion Auto-Encoder

In the last chapter, we exploited a new usage of the RGB image to guide supervised depth completion that only takes sparse depth as input, *i.e.*, incorporating it to the training loss. This is achieved by simultaneously recovering dense depth and reconstructing the image from the sparse input, and transferring image features to the depth branch. The proposed method can effectively reduce structure degradation in depth-only models, and significantly improve depth completion performance.

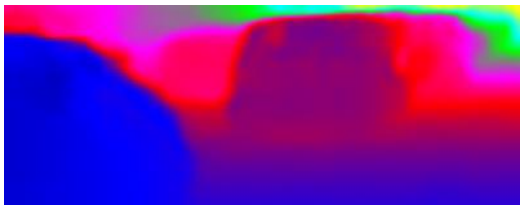
In this chapter, we focus on a more challenging task, *i.e.*, unsupervised depth completion only from sparse depth. Instead of resorting to the image as input and a second image for training like existing works, we propose to employ a single image to guide the learning process. This idea is inspired by the image guidance approach in the last chapter, but is more specific to the unsupervised setting. Specifically, we regard dense depth as a reconstructed result of the sparse input, and formulate our model as an auto-encoder. To reduce structure degradation resulting from sparse depth, we employ the image to guide latent features by penalizing their difference in the training process. The image guidance loss enables our model to acquire more dense and structural cues that are beneficial for producing more accurate and consistent depth values. For inference, our model only takes sparse depth as input and no image is required. Our paradigm is new and pushes unsupervised depth completion further than existing works that require the image at test time. On the KITTI Depth Completion Benchmark [Uhrig et al., 2017], we validate its effectiveness through extensive experiments and achieve good performance compared with other unsupervised works. The proposed method is also applicable to indoor scenes such as NYUv2 [Silberman et al., 2012].

In the reminder of this chapter, Section 6.1 introduces our motivation and main contributions. Section 6.2 reviews auto-encoders. Section 6.3 revisits existing unsupervised depth completion models. We illustrate the details of our method in Section 6.4 and validate its effectiveness through extensive experiments in Section 6.5. Section 6.6 summarizes the chapter and proposes the future work.

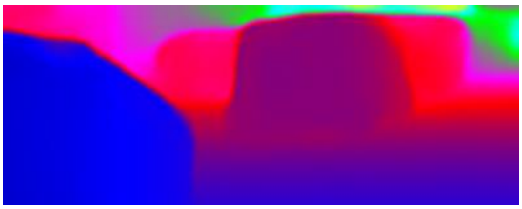
(a) RGB image



(b) Sparse depth



(c) S2D (image and depth as input)



(d) Ours (only depth as input)

Figure 6.1: Unsupervised depth completion from sparse depth. Compared with (a) the RGB image, object structures are more difficult to be identified and localized in (b) sparse depth due to too many missing depth values. (c) Existing unsupervised model S2D [Ma et al., 2019] takes the RGB image as an additional input. (d) Our model only inputs sparse depth. We achieve comparable performance to S2D in producing consistent depth values, especially around object boundaries, even without access to the image at test time.

## 6.1 Introduction

Unsupervised depth completion aims to recover dense depth from the sparse input without the supervision of dense ground truth. Compared with the supervised setting, unsupervised models do not involve expensive manual annotations.

In the depth completion community, a commonly acknowledged challenge is structure degradation, *i.e.*, object structures cannot be correctly localized and recovered [Lu et al., 2020; Qiu et al., 2019; Eldesokey, 2018]. Essentially, this problem is caused by the sparse nature of the input, *e.g.*, we can hardly tell where the car boundary is in Fig. 6.1(b) due to too many missing depth values. Fully-supervised models can reduce structure degradation by making use of dense ground truth, which provides per-pixel supervision[1] and covers most object structures. Many supervised works also address this problem by taking the RGB image as an extra input and fusing image features with sparse depth either through early or late fusion [Qiu et al., 2019; Van Gansbeke et al., 2019; Cheng et al., 2018; Jaritz et al., 2018].

For unsupervised depth completion, structure degradation becomes even harder to overcome because there is no dense ground truth available. Among the few works in the unsupervised setting, traditional non-learning (hand-crafted) methods [Kopf et al., 2007; Silberman et al., 2012; Barron and Poole, 2016; Ferstl et al., 2013] use hand-crafted matrix interpolation operations to fill in missing values, but lack effective

[1] In real practice, purely dense depth ground truth is difficult to acquire. Per-pixel supervision is not guaranteed in real-word datasets like KITTI [Uhrig et al., 2017] and NYU [Silberman et al., 2012].

image guidance. Recently, an alternative practice is to make use of network training, *i.e.*, taking the RGB image as an additional input and calculating the image warping loss either from stereo [Yang et al., 2019] or adjacent video frames [Ma et al., 2019; Wong et al., 2020, 2021a]. Clearly, compared with supervised methods where plain early and late fusion strategies are readily available, there are far fewer options for integrating image features in the unsupervised community.

In this chapter, we propose a new approach to integrating image features in unsupervised depth completion. In a nutshell, our method is formulated as an auto-encoder [Hinton and Zemel, 1993; Hinton et al., 2006], where sparse depth is first transformed into latent features and then recovered into dense depth. The sparse input serves as a supervision signal for the network. Besides the lack of structural cues, a vanilla auto-encoder will not give good performance on depth completion due to its trivial nature, *i.e.*, generating a trivial mapping from input to output as the input is also used for supervision. To improve performance, we employ an image to guide latent features during training, as illustrated in Fig. 1.7(b). In addition to providing dense structural cues, the image guidance constrains latent features to reduce the trivial solution. We show that this practice yields a large improvement over the vanilla baseline and allows our method to be competitive on public benchmarks.

We emphasize two distinctive characteristics of our design which make it novel and insightful. First, our method introduces a new setting in unsupervised depth completion, *i.e.*, only using sparse depth as the network input in both training and testing. In comparison, previous unsupervised works assume structure degradation can only be reduced by adopting the image and sparse depth at both training and test phases. We demonstrate the feasibility of this new setting with effective reduction of structure degradation and satisfying depth completion accuracy, which benefits the scientific body of literature in this area. Second, we provide insights on the appropriate use of image guidance through various studies, such as the position where image guidance is imposed, and the impact of feature resolution and channels.

In summary, we make the following major contributions:

- We propose a new paradigm for unsupervised depth completion that recovers dense depth only from the sparse input in both training and testing. We push this task further beyond existing unsupervised works that take the image as an additional input and employ a second image for training.

- Our method is formulated as an auto-encoder and uses the image to directly guide latent features in training. This enables our model to acquire more dense and structural cues, which improve the depth completion accuracy and reduce structure degradation even without the image input.

- We validate the effectiveness of the proposed image guidance and achieve good performance on the KITTI Depth Completion Benchmark compared with other unsupervised methods. Our model is also applicable to indoor scenes, *e.g.*, NYUv2.

## 6.2 Related Work

The related work on supervised depth completion is reviewed in Section 2.2.2, and unsupervised studies are introduced in Section 2.2.3. In this section, we give additional review on auto-encoders as our model takes the auto-encoder framework.

**Conventional auto-encoders.** Auto-encoders aim to generate a compressed feature representation by learning an identity mapping from the input to the output [Hinton et al., 2006]. The input itself is used as a supervision signal for the training process. Auto-encoders have been widely employed as an unsupervised learning technique in image denoising [Vincent et al., 2010, 2008], super-resolution [Rong et al., 2018; Zeng et al., 2015], multi-view learning [Wang et al., 2016], *etc*.

**Constrained auto-encoders.** The limitation of conventional auto-encoders is that in many cases, encoders cannot consistently extract discriminative and useful features. In that case, some irrelevant information may be retained [Wang and Ding, 2017]. To deal with this issue, the sparse auto-encoder is proposed to enhance the discrimination ability by constraining the output from hidden layers to a small value, *e.g.*, zero [Coates et al., 2011]. Other useful constraints include graph embeddings [Yu et al., 2013], non-negativity [Hosseini-Asl et al., 2016; Teng et al., 2019], label consistency [Hu et al., 2018], hierarchical feature selection [Masci et al., 2011], *etc*.

**Guided auto-encoders.** In addition to incorporating constraints, guiding latent features with a certain signal, known as guided auto-encoders [Bengio et al., 2007], is another useful strategy. The key idea is to add a supervised loss to the latent representation as guidance [Le et al., 2018]. This guidance encourages the network to acquire more relevant features in latent space (also referred to as latent representation disentanglement learning [Ding et al., 2020]), which is beneficial for the decoder to generate more satisfactory output. In the literature, those signals used for guidance include label information [Snoek et al., 2012], pose estimation [Li and Ji, 2020], feature selection [Wang and Ding, 2017], and so on. Our work uses the image to guide latent features, which supplies more structural cues to depth.

## 6.3 Unsupervised Depth Completion Revisited

Unsupervised depth completion models assume there is no dense ground truth or any other manual annotations available. To reduce structure degradation resulting from the sparse input $\mathbf{d} \in \mathbb{R}^{H \times W}$ ($H$ and $W$ represent the height and width respectively), existing studies [Ma et al., 2019; Yang et al., 2019; Wong et al., 2020, 2021a] further assume an associated RGB image $\mathbf{r} \in \mathbb{R}^{H \times W \times C}$ is available ($C$ is the number of channels, *e.g.*, 3 for an RGB image and 1 for its grayscale), and take it as an additional input, *i.e.*,

$$\tilde{\mathbf{d}} = f(\mathbf{d}, \mathbf{r}), \tag{6.1}$$

where $\tilde{\mathbf{d}} \in \mathbb{R}^{H \times W}$ is dense depth output. In this formulation, the sparse input is used as a supervision signal for depth, *i.e.*,

$$\ell_d = \frac{1}{N_1} \left\| \mathbf{M} \odot (\mathbf{d} - \tilde{\mathbf{d}}) \right\|^{\eta}, \tag{6.2}$$

where $\mathbf{M}$ is a binary mask that indicates validness of input depth (1 for points with depth values and 0 for none), and $N_1$ is the total number of valid points. $\eta$ is the norm of the loss, *i.e.*, 1 for $\ell_1$ (MAE) and 2 for $\ell_2$ (MSE). $\odot$ denotes element-wise multiplication. Additionally, a second image, either from stereo or adjacent frames, is employed to construct the disparity loss [Yang et al., 2019] or photometric loss [Ma et al., 2019; Wong et al., 2020, 2021a]. This loss is essentially implicit supervision to depth since it does not directly penalize depth reconstruction but the result derived from depth, *i.e.*, the warped image. Without loss of generality, we denote the additional loss as $\ell_c$, and thus the entire training loss $\ell_t$ becomes

$$\ell_t = \ell_d + w_c \cdot \ell_c, \tag{6.3}$$

where $w_c$ controls the impact of $\ell_c$. At test time, the second image is not required, but the image associated with sparse depth is still taken as input.

In addition, the models in [Yang et al., 2019; Wong et al., 2021a] have to learn prior information, *e.g.*, dense depth prior or topology prior, with another network pre-trained on the Virtual KITTI dataset [Gaidon et al., 2016]. Ma et al. [2019] compute feature correspondences from adjacent images for pose estimation, similar to [Wong et al., 2020]. These operations heavily rely on RGB images and other image related information, which are less practical in real-world applications. Also, the use of additional resources, *e.g.*, extra image, dataset or technique, further indicates the difficulty in integrating image features in unsupervised depth completion models.

## 6.4 Our Method

Section 6.3 motivates us to think about an easier but effective usage of RGB images, in which case structure degradation can still be reduced even without the image input. To this end, we formulate our model as an auto-encoder and propose to guide latent features with the image. The general framework is illustrated in Fig. 6.2. This approach generally has two distinctions: (1) It enables our model to recover dense depth *only* from the sparse input, which is a normal setting in supervised works [Uhrig et al., 2017; Chodosh et al., 2018; Ma et al., 2019; Eldesokey et al., 2019] but has not been well studied in the unsupervised area; (2) It is effective in better reducing structure degradation than using an auto-encoder without image guidance.

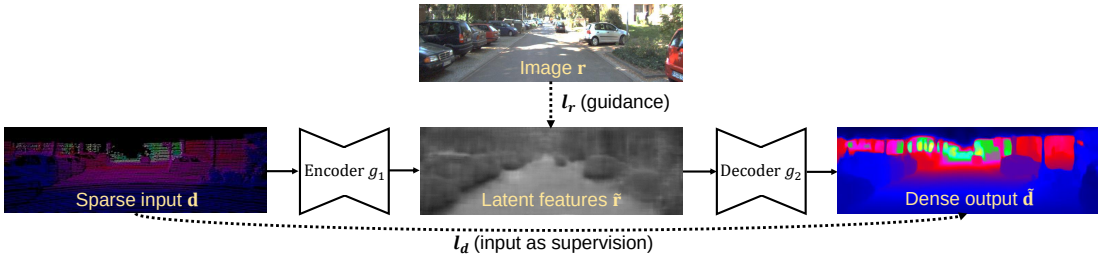



Figure 6.2: Proposed auto-encoder framework for training unsupervised depth completion. The encoder transforms sparse depth input into latent features, which are then fed into the decoder to produce dense depth. The sparse input itself is used as the supervision signal for training. In the figure, the latent feature map is obtained from our default model (see Section 6.5.1) and visualized by normalizing the values into 0-1.

### 6.4.1 Depth Completion as an Auto-Encoder

We aim to construct a model $g$ that recovers dense depth only from the sparse input, *i.e.*,

$$\tilde{\mathbf{d}} = g(\mathbf{d}). \tag{6.4}$$

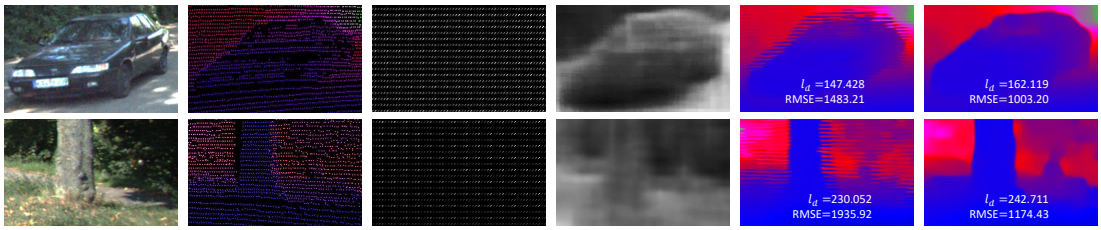
To achieve this, we regard the dense output as a reconstructed result of the sparse input, and formulate $g$ as an auto-encoder [Hinton and Zemel, 1993; Hinton et al., 2006] to realize this reconstruction. More specifically, we divide $g$ into an encoder $g_1$ and a decoder $g_2$. $g_1$ transforms the sparse input $\mathbf{d}$ into latent features $\tilde{\mathbf{r}} \in \mathbb{R}^{H_1 \times W_1 \times C_1}$ where $H_1$ and $W_1$ represent the height and width, and $C_1$ is the number of feature channels. $g_2$ recovers dense depth from $\tilde{\mathbf{r}}$. The entire process is described as:

$$\tilde{\mathbf{d}} = g(\mathbf{d}) \rightarrow \tilde{\mathbf{d}} = g_2(\tilde{\mathbf{r}} = g_1(\mathbf{d})). \tag{6.5}$$

The model can be trained with the identity mapping loss defined in Eq. 6.2. We name this model the *vanilla auto-encoder* because it does not incorporate any extra information. Below, we list two major problems with the vanilla auto-encoder.

**Insufficient structural cues.** Without additional guidance, *e.g.*, the image, both the sparse input and its latent features cannot provide sufficient structural cues for accurate depth completion, particularly around object boundaries. For example, in Fig. 6.3(c), the latent features of the sparse input are still highly sparse, and we can hardly find any clear and useful structural information of the car and tree from them. The completed results based on these features present inconsistent depth values around boundaries (see Fig. 6.3(e)). Hence, it is difficult to recover consistent and accurate dense depth only from the sparse input.

**Trivial solution.** $g$ takes the sparse input $\mathbf{d}$ as both input and supervision, which may produce a trivial solution that $\tilde{\mathbf{d}}$ is infinitely close to $\mathbf{d}$ in valid positions that contain input values. The accuracy of other missing values to be completed is largely sacrificed. As shown in Fig. 6.3(e) and (f), even though the difference between the output and the sparse input is smaller with the vanilla model, the errors, *i.e.*, RMSE,




(a) RGB image (b) Sparse input (c) Vanilla latent features (d) Our latent features (e) Vanilla depth (f) Our depth

Figure 6.3: Comparison between vanilla and our image guided auto-encoders. (a) and (b) are the RGB image (not used as input) and the sparse input. (c) Vanilla latent features directly from sparse depth are also highly sparse, and they cannot indicate any clear or useful structural information. (d) Our image-guided latent features, by contrast, are able to acquire more dense and structural cues, *e.g.*, the general shapes of the car and tree are clearer than (c). (e) Dense depth from the vanilla auto-encoder fails to complete object boundaries properly. It has a smaller difference $\ell_d$ to the input, but larger errors compared with ground truth. (f) Our depth with guided latent features produces more visually consistent boundaries and more accurate depth values. This also indicates the reduced impact of the trivial solution as $\ell_d$ is slightly larger, but the RMSE is much smaller. The latent feature map is obtained from our default model (see Section 6.5.1) and visualized by normalizing the values into 0-1.

are larger. This can also be reflected by visual results, where stripe artifacts with similar patterns to horizontally scanned LiDAR points, exist around object boundaries. The negative impact of the trivial solution should be reduced.

### 6.4.2 Image Guidance to Latent Features

To deal with above issues, we propose to use the image to guide latent features in the training process (see Fig. 6.2). It aims to regularize latent features to obtain more structural cues from the image and reduce the trivial solution. We define a function $\phi$ that converts the image $\mathbf{r} \in \mathbb{R}^{H\times W\times C}$ into the image feature representation $\phi(\mathbf{r}) \in \mathbb{R}^{H_1\times W_1\times C_1}$ that shares the identical feature sizes with the latent features $\tilde{\mathbf{r}}$. $\phi$ is either (1) a self mapping, *i.e.*, $\phi(\mathbf{r}) = \mathbf{r}$, or (2) a CNN to extract convolutional features. The guidance works by penalizing the difference between the two features, *i.e.*,

$$\ell_r = \frac{1}{H_1 W_1 C_1} \left\| \phi(\mathbf{r}) - \tilde{\mathbf{r}} \right\|^{\gamma}, \tag{6.6}$$

where $\gamma$ determines the norm of the loss. Combined with the sparse depth loss defined in Eq. 6.2, the total training loss of the proposed model is

$$\ell_{total} = \ell_d + w_r \cdot \ell_r, \tag{6.7}$$

where $w_r$ weighs the impact of the image guidance loss $\ell_r$. Both the encoder and decoder share the same U-shaped architecture, *i.e.*, simplified SegNet [Badrinarayanan

et al., 2017] with 14 convolutional layers (pooling layers are removed). It gradually downsamples feature maps to 1/16 of the input resolution and upsamples them to produce the full-sized output.

### 6.4.3 Discussion

**Why can dense depth be directly constructed from sparse input only?** For a specific position in sparse depth, convolving with a squared kernel is like performing a weighted sum within the local region. If that position has no depth value, it will be updated based on nearby points with values. This is the underlying reason that dense depth can be directly constructed from the sparse input. Note that supervised by valid points in sparse depth, the weights are learnable. This is an advantage over traditional hand-crafted methods such as nearest neighboring and bilinear interpolation. We will later show through experiments that our model can produce more accurate results than these methods.

**The role of image guidance.** Image guidance enables latent features to better acquire dense and structural cues that can facilitate depth completion. In Fig. 6.3(d), guided latent features of the car and tree reveal their general shapes more clearly than vanilla (unguided) ones that only have sparse representations. As illustrated in Fig. 6.3(f), both examples have a larger $\ell_d$ but lower RMSE than vanilla results, indicating the reduced impact of the trivial solution.

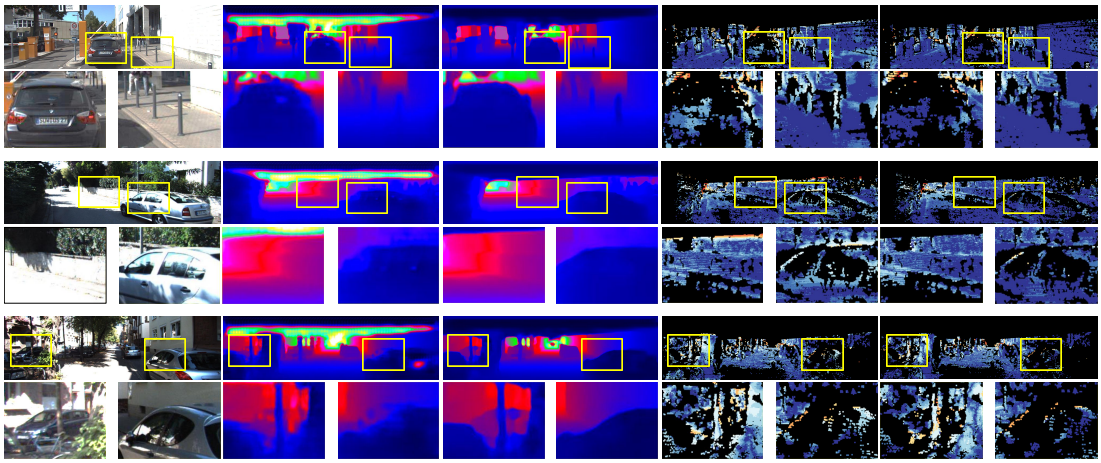
In fact, the proposed image guidance is inspired by the work in Chapter 5, which acquires image features by reconstructing the image from sparse depth [Lu et al., 2020]. The similarity with ours is that both works add the image loss as part of the training loss. However, the underlying insights of such image guidance are different. In terms of the network architecture, our method does not have an image decoder separate from the main branch, so it is not aimed for image reconstruction. Functionally, our image guidance is directly imposed to latent depth features by penalizing their difference (regarded as *explicit guidance*), and dense depth has to be recovered from the refined features. By contrast, in the previous chapter, we implicitly generate image-related features by reconstructing the image as an output. We will justify the effectiveness of our method in the unsupervised setting through experiments.

**Relationship with existing unsupervised models.** Our formulation for training the model in Eq. 6.7 is consistent with the general form of the unsupervised framework defined in Eq. 6.3. The image guidance loss $\ell_r$, similar to $\ell_c$ in Eq. 6.3, is an extra loss that facilitates network training. However, it is essentially different from other unsupervised works [Ma et al., 2019; Yang et al., 2019; Wong et al., 2020, 2021a] in that (1) it focuses on enhancing intermediate latent features, and (2) it does not require a second image for training.

**Inference.** Learning the proposed depth completion auto-encoder only requires the image during training. At test time, our model only takes sparse depth as input (see Fig. 1.7(b)), *i.e.*,

$$\tilde{\mathbf{d}} = g(\mathbf{d}; \theta_g^*), \tag{6.8}$$

where $\theta_g^*$ denotes the parameters of the optimal model.



(a) RGB image (b) IR* depth (c) Our depth (d) IR* error map (e) Our error map

Figure 6.4: Qualitative comparison with IR* [Lu et al., 2020] proposed in the previous chapter. It is retrained with the unsupervised setting, *i.e.*, using the input as supervision. Our model outperforms it in some key regions such as object boundaries with smaller errors.

## 6.5 Experiments

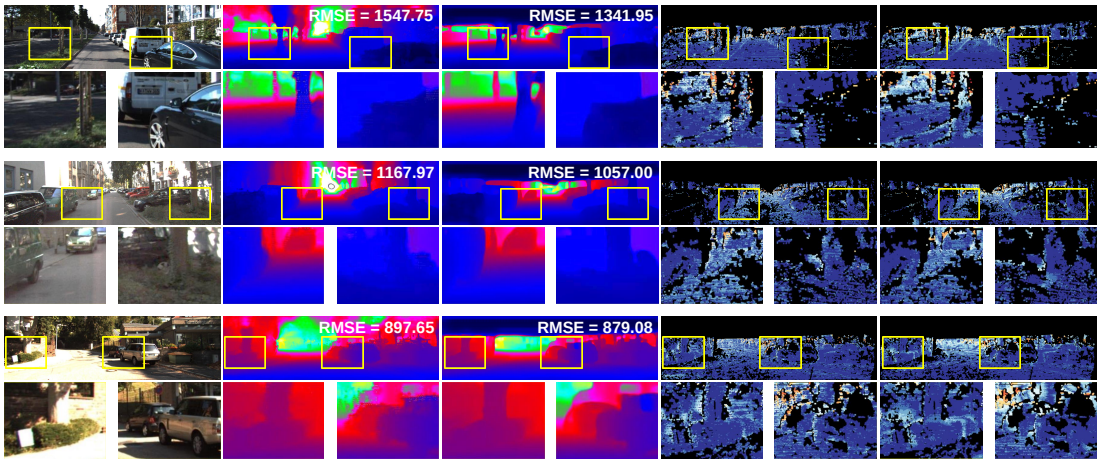
In this section, we demonstrate the effectiveness of our method through both quantitative and qualitative results.

### 6.5.1 Implementation Details

**Dataset.** We report depth completion results on the KITTI Depth Completion Benchmark [Uhrig et al., 2017]. The KITTI depth maps are acquired by reprojecting LiDAR points taken over a short time window onto an image, and around 5% of the pixels have depth values. When counting sparse depth maps, there are 85,898 training, 1,000 validation, and 1,000 test images in total. Ground truth depth maps are generated by accumulating LiDAR points from adjacent frames using semi-global matching, with outliers manually removed [Uhrig et al., 2017]. Test set results are evaluated on the online benchmark server with no ground truth available to the public.

**Evaluation metrics.** Following the benchmark [Uhrig et al., 2017], we use four quantitative evaluation metrics: (1) root mean square error (RMSE in mm), (2) mean absolute error (MAE in mm), (3) RMSE of inverse depth (iRMSE in 1/km), and (4) MAE of inverse depth (iMAE in 1/km). Among them, RMSE is used to rank approaches on the benchmark.

**Training procedure.** We implement our network with PyTorch [Paszke et al., 2017], and train and test the model on one NVIDIA Titan X GPU. All the training data have a resolution of $352 \times 1216$. The model is trained with the Adam optimizer [Kingma and Ba, 2014], where the initial learning rate is set as 0.001. In our default model, $\eta = 1$, $\gamma = 1$, $w_r = 0.1$. Besides, latent features $\tilde{\mathbf{r}} \in \mathbb{R}^{352\times1216\times1}$ share the same spatial resolution, *i.e.*, height and width, with the input. Also, they only




(a) RGB image (b) S2D depth (c) Our depth (d) S2D error map (e) Our error map

Figure 6.5: Qualitative comparison with the unsupervised learning model S2D [Ma et al., 2019] on the KITTI *test* set. S2D inputs RGB images at both training and test phases and employs a second image during training for implicit depth supervision. Our RMSE in three examples is better than S2D, and our results present smaller errors in some challenging regions, *e.g.*, car boundaries and poles.

have one feature channel that is directly guided by the gray-scale image without any convolutional layers to extract image features, *i.e.*, $\phi(\mathbf{r}) = \mathbf{r} \in \mathbb{R}^{352\times1216\times1}$. The default vanilla auto-encoder has identical latent feature resolution and number of feature channels to our default model.

### 6.5.2 Comparison with Existing Methods

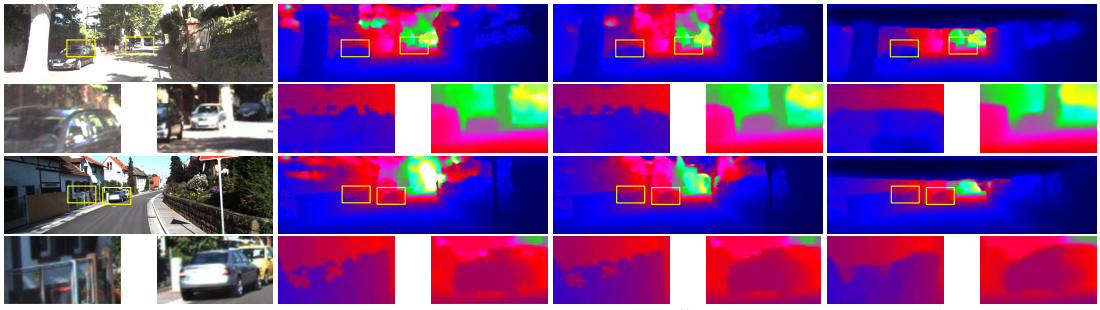
We compare four published unsupervised works, S2D [Ma et al., 2019], DDP [Yang et al., 2019], VOICED [Wong et al., 2020], and ScaffFusion [Wong et al., 2021a]. Note that although IR [Lu et al., 2020] introduced in the previous chapter is not specially designed for unsupervised depth completion, its usage of RGB images is similar to the proposed unsupervised model, *i.e.*, incorporating the image loss in training. For a fair comparison, we retrain IR by replacing dense ground truth with sparse depth. To distinguish it from the original fully-supervised model, we rename it as IR*. We additionally compare several hand-crafted methods.

**IR*.** We report quantitative results in Table 6.1. Our model significantly outperforms IR* [Lu et al., 2020], *i.e.*, surpassing RMSE by 491.61 (25.3%), MAE by 111.62 (20.6%), iRMSE by 12.99 (72.65%), and iMAE by 5.98 (77.1%). Qualitative results in Fig. 6.4 also indicate the superiority of our model in key regions such as object boundaries.

The primary reason for our superior performance over IR* is that the proposed image guidance gives direct and explicit refinement to latent features. This can be regarded as a "brute-force" or "passive" refinement because we directly penalize their difference. By contrast, IR* implicitly learns image-related features by reconstructing the image from sparse depth. There is no guarantee that image features can be prop-

| | Method | #Param. | RMSE ↓ | MAE ↓ | iRMSE ↓ | iMAE ↓ |
|---|---|---|---|---|---|---|
| **Unsupervised** (only sparse input) | IR* | 11.63M | 1943.28 | 541.36 | 17.88 | 7.76 |
| | Ours | 2.29M | 1451.67 | 429.74 | 4.89 | 1.78 |
| **Unsupervised** (sparse & RGB inputs) | S2D | 18.8M | 1299.85 | 350.32 | 4.07 | 1.57 |
| | DDP | 27.8M | 1263.19 | 343.46 | 3.58 | 1.32 |
| | VOICED | 9.7M | 1169.97 | 299.41 | 3.56 | 1.20 |
| | ScaffFusion | 7.8M | 1121.89 | 282.86 | 3.32 | 1.17 |

Table 6.1: Quantitative comparison with unsupervised methods on the KITTI *test* set. The methods we compare include IR* [Lu et al., 2020] (this method is retrained with the unsupervised setting, *i.e.*, replacing dense ground truth with input sparse depth), S2D [Ma et al., 2019], DDP [Yang et al., 2019], VOICED [Wong et al., 2020], and ScaffFusion [Wong et al., 2021a]. These results are calculated from the benchmark server, and no ground truth is available to the public. ↓ means smaller is better.



(a) RGB image (b) VOICED (c) ScaffFusion (d) Ours

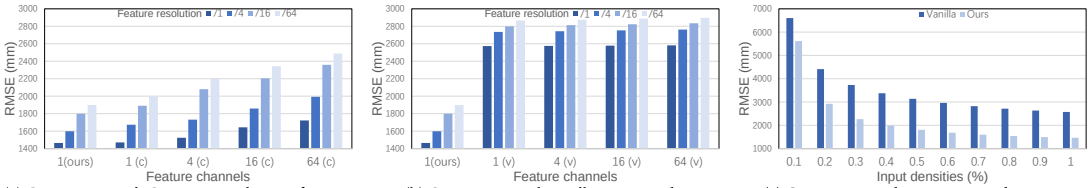
Figure 6.6: Qualitative comparison with VOICED [Wong et al., 2020] and ScaffFusion [Wong et al., 2021a] on the KITTI *test* set. Our model produces more visually-consistent depth values.

erly transferred to the depth branch. This is because (1) there is no penalty between depth and image features, and (2) it is more difficult for depth features to coincide with image features with such limited depth points for supervision. Another reason is a too complicated network, *i.e.*, IR* has nearly 5 times more parameters than ours, does not necessarily yield good performance in the auto-encoder framework (IR* also works like an auto-encoder when trained with sparse depth). This issue will be further discussed in Section 6.5.3.

**S2D, DDP, VOICED, and ScaffFusion.** Quantitative results on the KITTI test set are reported in Table 6.1. Naturally, our method does not beat the four works in numbers due to the input difference and less additional information used during training. Even so, we still achieve competitive performance in some visual examples.

We provide qualitative comparison with S2D [Ma et al., 2019], including the output dense depth and its error maps[2], in Fig. 6.5. Compared with S2D [Ma et al.,

[2]These error maps are copied directly from the benchmark. Lighter regions correspond to larger errors. Note that we did not find the unsupervised results of DDP [Yang et al., 2019] from the benchmark, and no code was published. Hence, there is no qualitative comparison with it.




(a) Comparison with CNN-extracted image features (b) Comparison with vanilla auto-encoder (c) Comparison with various input densities

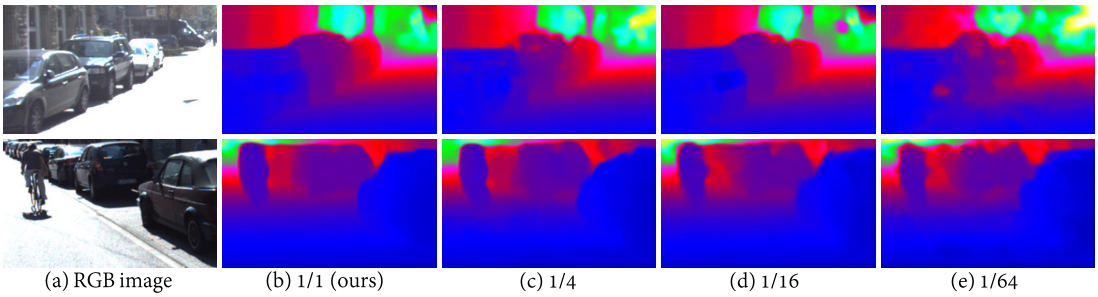
Figure 6.7: Model analysis on the KITTI *validation* set. (a) Impact of the resolution and number of channels of latent features. "c" means that we use CNNs to extract the image features. (b) Comparison with the vanilla auto-encoder with different feature resolutions and channels. "v" represents the vanilla auto-encoder. (c) Robustness to input densities. Here the vanilla auto-encoder share the same latent feature resolution and channel with our default image guided model.

2019], we demonstrate competitive performance in producing accurate depth results in some challenging regions that have few input sparse depth values, such as object boundaries (*e.g.*, the cars in three examples and the tree in the third example) and small objects (*e.g.*, the poles in the first two examples). It reveals that our model can reduce structure degradation even without the image as input at test time. Besides, we achieve a smaller RMSE in all the three examples than S2D [Ma et al., 2019]. We additionally compare VOICED [Wong et al., 2020] and ScaffFusion [Wong et al., 2021a] in Fig. 6.6. Our model can produce more consistent object structures than the two works.

**Hand-crafted methods.** We give quantitative comparison with hand-crafted methods on the KITTI validation set in Table 6.2. Overall, our performance is significantly better than all the hand-crafted methods in terms of RMSE and MAE. Among them, traditional interpolation methods, *i.e.*, nearest, bilinear, and bicubic, have much larger errors because these interpolation approaches only perform a naive local operation based on available depth values. Other methods, *i.e.*, TGV [Ferstl et al., 2013], Bilateral [Silberman et al., 2012], Fast [Barron and Poole, 2016], are based on hand-crafted features that are less informative and robust than learnable features in our model (see Section 6.4.3). Thus, they are much less comparable to our performance even with RGB as input.

### 6.5.3 Model Analysis & Ablation Studies

**Impact of the resolution and number of channels of latent features.** For clarity, the feature resolution refers to the spatial dimension, *i.e.*, height and width, and channels represent the number of feature maps. We first investigate their impact to our image-guided model. For the self-mapping, we set the latent channel number as 1, and then use the one-channel gray-scale image to directly guide latent features. For the CNN mapping, we apply a 3-layer convolutional network with $3 \times 3$ kernels to extract image features from the image. From Fig. 6.7(a), we observe that using CNNs to extract image features does not bring significant performance gain, *i.e.*, using the original image to directly guide latent features yields the best RMSE in all feature



(a) RGB image (b) 1/1 (ours) (c) 1/4 (d) 1/16 (e) 1/64

Figure 6.8: Depth completion with different resolutions of latent features. Compared with our default model that retains the full resolution of latent features, reducing the resolution leads to less consistent and accurate depth completion results. The number of feature channels in all cases is 1, *i.e.*, our default setting.

resolutions. Moreover, adding extra channels with CNNs reduces performance. The degraded results can also be found in the vanilla auto-encoder (see Fig. 6.7(b)). We attribute the performance degradation after using more parameters to the *over-complete auto-encoder* [Vincent et al., 2010], *i.e.*, the model tends to simply copy the input to the output without learning useful features. This problem is primarily caused by having excessive parameters in the hidden layer, *e.g.*, using too many channels or a very complex network. In our system, the sparse input only has one channel, so we design latent features to have one channel and the network to be light-weight. Experimental results have validated the effectiveness of our design.

For the feature resolution, we find that reducing the height and width leads to larger errors. This is because the spatial correspondence at each position between the input, image, and output cannot be well preserved with the reduced resolution. This can also be reflected in visual results, as illustrated in Fig. 6.8. Reducing the feature resolution yields worse results, especially in regions that require strong spatial correspondence between depth and the image, *e.g.*, the car boundary.

**Effectiveness of image guidance over the vanilla auto-encoder.** To validate the effectiveness of image guidance, we compare the vanilla auto-encoder and our image-guided model. In Fig. 6.7(b), our model outperforms the vanilla one to a large margin with various settings of feature resolutions and channels. Besides, combining Fig. 6.7(a) and (b), the overall performance after using image guidance is significantly better than the vanilla model in all cases. For quantitative results in Table 6.2, our model surpasses its RMSE by 1108.02 (43.1%) and MAE by 265.3 (38.1%). We give visual comparison in Fig. 6.9. Our image guidance is indeed effective in improving depth completion accuracy.

**Reasons for intermediate guidance.** To verify intermediate guidance to latent features, we place this guidance to the encoder (EG), the decoder (DG), and output (OG) respectively, and then retrain the model. From Table 6.2, employing image guidance to the encoder produces slightly worse results, which is because depth features from the sparse input have not been sufficiently encoded. By contrast, the performance after moving the guidance to the decoder is significantly degraded. The

| | Method | RMSE ↓ | MAE ↓ |
|---|---|---|---|
| **Unsupervised** (only sparse input) | Vanilla | 2572.71 | 696.53 |
| | Ours | 1464.69 | 431.23 |
| | EG | 1985.38 | 583.01 |
| | DG | 2496.80 | 682.19 |
| | OG | 2723.14 | 711.77 |
| **Hand-crafted** (only sparse input) | Nearest | 4268.83 | 1934.50 |
| | Bilinear | 3844.69 | 1820.03 |
| | Bicubic | 3821.33 | 1779.58 |
| **Hand-crafted** (sparse & RGB inputs) | TGV | 2761.29 | 1068.69 |
| | Bilateral | 2989.02 | 1200.56 |
| | Fast | 3548.87 | 1767.80 |

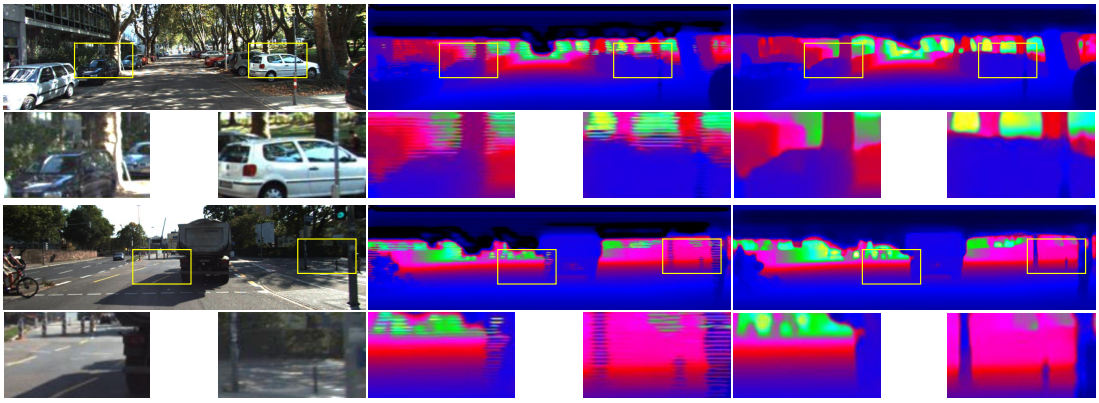
Table 6.2: Quantitative comparison with the vanilla auto-encoder, different positions of image guidance, and hand-crafted methods on the KITTI *validation* set. "EG", "DG", and "OG" mean image guidance is placed to the encoder, decoder, and output respectively. In addition to simple interpolation methods, *i.e.*, nearest, bilinear, and bicubic, we also compare TGV [Ferstl et al., 2013], Bilateral [Silberman et al., 2012], and Fast [Barron and Poole, 2016]. ↓ means smaller is better.

underlying reason is the decoder's task is to recover dense depth from latent features. Requiring an additional task of the decoder detracts from the depth completion task. An extreme case is after we place image guidance at the output layer, the results become even worse than the vanilla model (the decoder has to both recover dense depth and reconstruct the image, which is difficult to work well). In conclusion, applying image guidance to intermediate latent features yields the best results, where the original depth features have been well encoded and refined, and the decoder can focus on depth completion.

**RGB *vs.* gray guidance.** We can use either the RGB image or its grayscale ($gray = 0.3 \times R + 0.59 \times G + 0.11 \times B$, where $R$, $G$, and $B$ represent three channels of the RGB image respectively) to guide latent features. They do not differ too much in terms of the final performance because the image content in two color space is similar, *e.g.*, important structures contained in RGB are also mostly visible in gray. Fig. 6.10 shows the RGB image (c) and its gray image (d), as well as their guided latent features (e and f) and corresponding dense depth (m and n). We do not see obvious difference between two types of latent features and dense output, except that latent features guided by the gray image present brighter appearance. According to the quantitative results in Table 6.3, we find that using the gray image for guidance yields slightly better results. This is because it is harder to penalize RGB and latent features as it involves three channels. Also, more feature channels make the model easier to be affected by the over-complete issue (see above). In fact, in Fig. 6.10(e), we show that the learned 3-channel latent feature (visualized like an RGB image) does not present obvious colors, *i.e.*, it still looks gray. It suggests that the network does not rely on specific colors for completion. Structure information indicated by intensity difference is more important. A similar phenomenon on better results with the gray image is

| | Method | RMSE ↓ | MAE ↓ |
|---|---|---|---|
| **Guiding LF** | RGB | 1485.85 | 439.76 |
| | Gray | 1464.69 | 431.23 |
| **Replacing LF** (w/o retraining) | RGB | 24499.30 | 9209.43 |
| | Gray | 16107.76 | 8867.21 |
| **Replacing LF** (with retraining) | RGB | 4615.71 | 2003.55 |
| | Gray | 5134.19 | 2321.09 |

Table 6.3: Quantitative results of using RGB and gray images to guide or replace latent features on the KITTI *validation* set. ↓ means smaller is better. There is not significant difference between using the RGB or gray image to guide latent features. Replacing latent features with the image, either with or without retraining, produces poor results. "LF" represents latent features.



(a) RGB image (not used) (b) Vanilla depth (c) Our depth

Figure 6.9: Qualitative comparison with the vanilla auto-encoder on the KITTI *test* set. Our model significantly outperforms it in producing more consistent and accurate depth values.

observed in [Ma et al., 2018; Lu et al., 2020]. Moreover, in terms of practical use, the gray image occupies less storage than RGB. Hence, by default, we use the gray image to guide latent features.

**Replacing latent features with the image.** The image guidance loss defined in Eq. 6.6 facilitates latent features with more structural cues beneficial for depth completion. It is achieved by penalizing the difference between latent features and the image. A natural question is, what will the performance be if we replace latent features with the image, *i.e.*, the image guidance loss is equal to zero?

The first experiment is, given our trained model with default settings, we replace latent features with the image directly at test time. As shown in Fig. 6.10(g) and (h), the decoder cannot recover any correct depth, which is also reflected by the extremely poor quantitative results in Table 6.3. The underlying reason is that the trained parameters are fixed, and the direct replacement destroys the learned information in latent layers. In that case, the decoder, originally having latent features as input,

| | Method | RMSE ↓ | REL ↓ |
|---|---|---|---|
| **Unsupervised** (only sparse input) | Vanilla | 0.449 | 0.081 |
| | IR* | 0.358 | 0.062 |
| | Ours | 0.315 | 0.053 |
| **Hand-crafted** (sparse & RGB inputs) | TGV | 0.635 | 0.123 |
| | Bilateral | 0.479 | 0.084 |

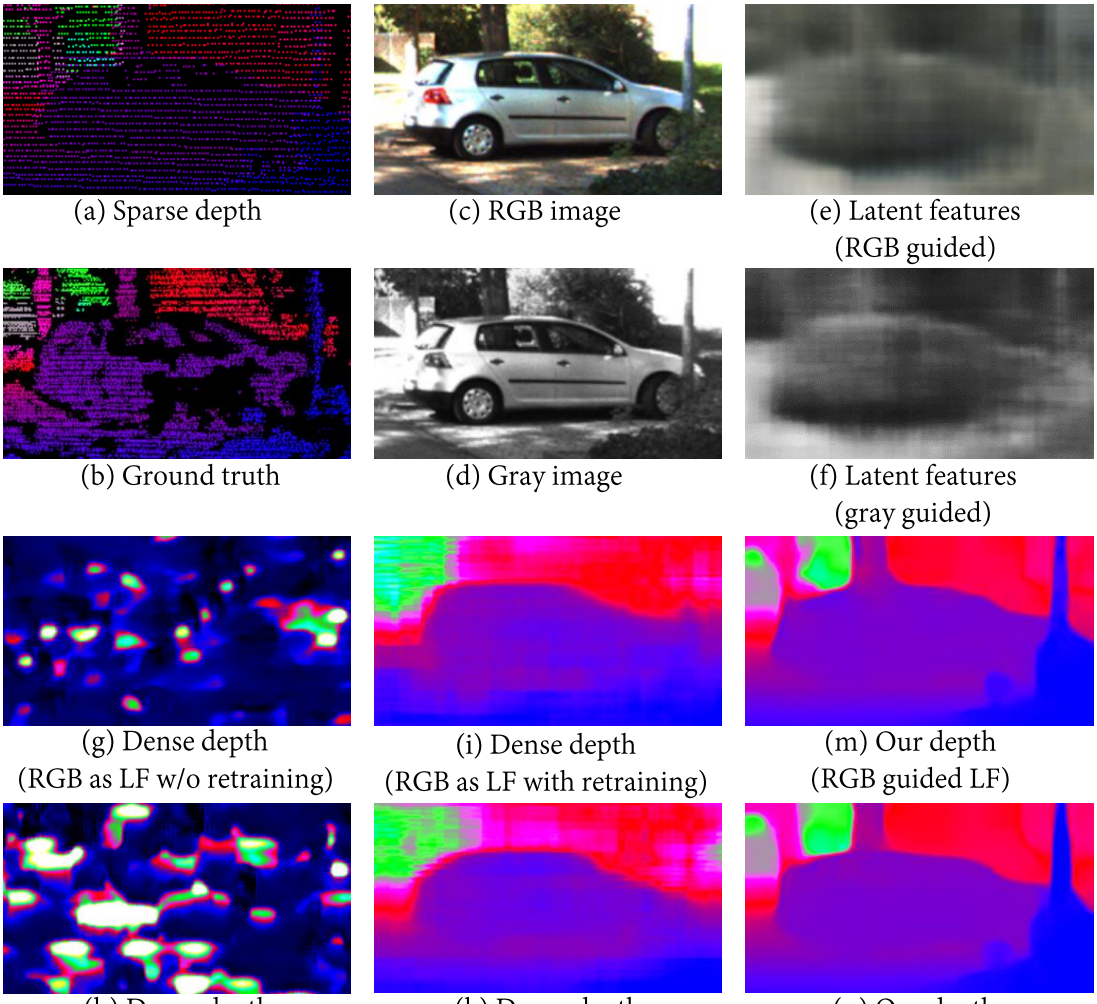
Table 6.4: Quantitative comparison with the vanilla auto-encoder and hand-crafted methods on the NYUv2 *test* set. ↓ means smaller is better. Here the vanilla auto-encoder share the same latent feature resolution and channel with our default image-guided model. Our model outperforms the vanilla auto-encoder, IR* [Lu et al., 2020], and hand-crafted methods (TGV [Ferstl et al., 2013] and Bilateral [Silberman et al., 2012]). It indicates that our method has good applicability to other dataset.

does not know how to extract useful features from the image.

The second experiment is that we replace latent features with the image and retrain the entire model. This makes more sense as it actively adjusts parameters. In that case, the encoder is blocked and the network becomes to use the decoder to recover dense depth directly from the image, *i.e.*, depth estimation from a single image supervised by the sparse input. We can observe in Fig. 6.10(i) and (k) as well as Table 6.3 that this approach produces better results than the model above without retraining. However, the performance is still less competitive than ours. Visual results indicate that the depth of some important details, *e.g.*, trees and car boundaries, cannot be properly recovered.

Based on these results, we find that replacing latent features with the image is less effective for depth completion. Specifically, latent features guided by the image and the image itself are two different concepts. Latent features are a type of feature representation of sparse depth. Guided by the image, they are embedded with more consistent structural cues, but are still conditioned on sparse depth rather than the image. By contrast, the image is another modality inherently different from depth. It cannot be regarded as a latent representation of sparse depth, so recovering depth directly from the image is less accurate (we refer the reader to the KITTI Depth Estimation Benchmark [Uhrig et al., 2017] where the overall accuracy is much worse than depth completion results, and the same conclusion is also drawn in [Liu et al., 2021; Qu et al., 2020]). In summary, guiding latent features with the image is more beneficial for depth completion than directly replacing them with the image.

**Robustness to input densities.** We also analyze the robustness to different input densities. The valid points with depth values in the original sparse input account for around 5% of the entire depth map. We further reduce the input sparsity by randomly retaining points with ratios from 90% to 10%. Our results in Fig. 6.7(c) demonstrate good robustness of our model to different densities (measured by RMSE). With an increased density, depth completion performance is gradually enhanced because more input data is provided. An extreme case is when the input data is too limited, *e.g.*, 10% remaining, our model presents much larger errors because there is



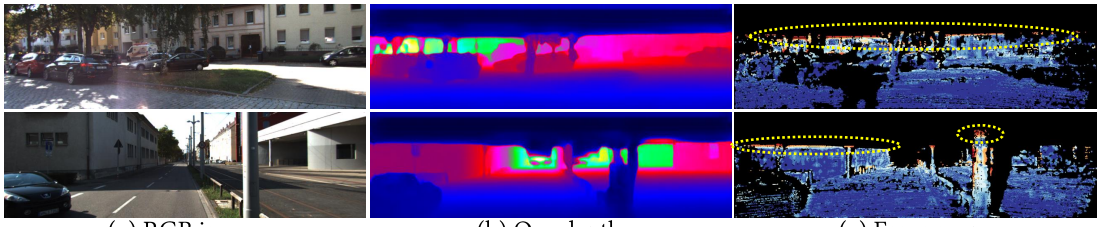
(a) Sparse depth (c) RGB image (e) Latent features (RGB guided)

(b) Ground truth (d) Gray image (f) Latent features (gray guided)

(g) Dense depth (RGB as LF w/o retraining) (i) Dense depth (RGB as LF with retraining) (m) Our depth (RGB guided LF)

(h) Dense depth (gray as LF w/o retraining) (k) Dense depth (gray as LF with retraining) (n) Our depth (gray guided LF)

Figure 6.10: Qualitative results of using RGB and gray images to guide or replace latent features. There is no significant difference between using the RGB or gray image to guide latent features. Replacing latent features with the image, either retraining or not retraining the model, cannot produce better results than ours. "LF" represents latent features.



(a) RGB image (b) Our depth (c) Error map

Figure 6.11: Challenge cases. When the height of ground truth depth is much higher than sparse depth, more errors will occur in upper regions.

little structure information acquirable and recoverable from the input. Moreover, our model performs consistently better in all cases than the vanilla auto-encoder, which further indicates its effectiveness.

**Application to Indoor Scenarios.** Our depth completion auto-encoder can also be applied to indoor scenes, *e.g.*, NYUv2 [Silberman et al., 2012]. Each sparse input for training has 500 randomly selected depth values, the same as [Lu et al., 2020; Qiu et al., 2019]. We evaluate the proposed model on the official labelled test set that contains 654 samples. In Table 6.4, we report RMSE and REL. Our model significantly outperforms the vanilla auto-encoder, IR* [Lu et al., 2020], and hand-crafted methods (TGV [Ferstl et al., 2013] and Bilateral [Silberman et al., 2012]).

**Challenging cases.** In some cases, the actual height of ground truth depth is much higher than sparse depth, *i.e.*, there are few or even no input values in upper regions. Our model cannot perform well on these unseen pixels due to the severe lack of depth information for completion. Hence, compared with ground truth, our model produces larger errors there (see Fig. 6.11). How to solve this problem can be a potential direction for future research.

## 6.6 Conclusion

In this chapter, we have proposed a new unsupervised depth completion model. Formulated as an auto-encoder, our model only takes sparse depth as input, which is essentially different from existing unsupervised works that use the RGB image as an additional input at both training and test phases. To reduce structure degradation, we have proposed to employ the image to guide latent features in the training process. This approach enables the acquisition of more dense and structural features beneficial for producing more consistent and accurate depth values. We have validated its effectiveness through extensive experiments on the KITTI Depth Completion Benchmark [Uhrig et al., 2017] and achieved competitive performance against competing approaches. We have also given insights on the appropriate use of image guidance in terms of the resolution and number of channels of latent features. Our method has good applicability to indoor scenes, *e.g.*, NYUv2 [Silberman et al., 2012]. Our future work will focus on leveraging other information, *e.g.*, 3D point clouds, surface normal, to enhance latent features.

Chapter 7

# Conclusion and Future Work

## 7.1 Conclusion

In this thesis, we have explored the use of guidance to reduce structure degradation in two typical image enhancement tasks, *i.e.*, image smoothing and depth completion. For image smoothing, we have found that single structure guidance based on gradients or intensity difference in existing methods is not robust enough to properly differentiate between structures and textures. To address this issue, we have introduced the novel concept of "texture guidance" that indicates the location and magnitude of textures. We have combined it with semantic structure guidance and equip the filter with robust "structure-awareness" and "texture-awareness". Experimental results have demonstrated that with the two forms of guidance, the filter is able to remove strong textures without degrading main structures. For depth completion, we aim to deal with the structure degradation issue in depth-only models that only take sparse depth as input. Instead of resorting to the RGB image as an extra input, we have incorporated it as part of the training loss. In the supervised model, we have treated image reconstruction from sparse depth as an auxiliary task and used the image to supervise the reconstruction process, where image features can be transferred to the depth completion branch. For the unsupervised model, we have used the image to guide latent features by penalizing their difference. Hence, more dense and structural information can be aggregated with depth features. This new usage of image guidance is beneficial for enhancing depth-only depth completion accuracy in both supervised and unsupervised settings, *e.g.*, recovering more semantically-consistent object boundaries and small/thin objects. In the following, we highlight primary advantages of each method.

**Double-guided filter (Chapter 3):** (1) It is the first kernel filter that simultaneously employs structure guidance and texture guidance. (2) It provides an appropriate use of structure guidance and texture guidance within a kernel, which is intuitive and effective. (3) It eliminates the negative effect of gradients or intensity difference and can better differentiate between structures and textures, which benefits removing strong textures without blurring main structures.

**Texture and structure aware filtering network (Chapter 4):** (1) It overcomes the natural shortcomings of hand-crafted features in poor discrimination of textures when they present various spatial and color variations. (2) It provides an approach

to generating synthetic data for training both texture prediction and image smoothing and evaluating existing smoothing methods. (3) The learned texture guidance effectively adapts to different types of textures and significantly improves smoothing results after being combined with semantic structure guidance.

**Supervised depth completion via auxiliary image reconstruction (Chapter 5):** (1) It exploits a new usage of the RGB image in supervised depth completion, *i.e.*, incorporating it into the training loss. (2) It significantly outperforms existing depth-only models in recovering more consistent boundaries and small/thin objects. (3) It only requires the image at the training stage, which is practical and easy to implement in real-world applications.

**Unsupervised depth completion auto-encoder (Chapter 6):** (1) It provides a new approach to integrating image features, which enriches the scientific body of literature in the unsupervised depth completion community. (2) It provides a large reduction in structure degradation in the unsupervised depth-only setting and achieves comparable performance with existing unsupervised works that take the image as an extra input and use a second image during training time. (3) It does not need any image in testing, which is more practical and implementable than other unsupervised methods.

Through the four research works, we have a better understanding of the underlying reasons for structure degradation and how to handle it with the proper definition and use of guidance. In the next section, we will discuss several potential directions for further research.

## 7.2 Future Work

We provide several potential directions below to extend our research in the future.

### 7.2.1 Image Smoothing

**(1) Incorporating more semantic cues**

As illustrated in Fig. 4.8, the eyes, nose and other semantically-important information are undesirably removed as textures because they have texture-like patterns, *e.g.*, small-scale dotted appearance. We can hardly recognize the person after they are removed, which deviates from the essential definition of image smoothing, *i.e.*, only smoothing out insignificant details. To deal with this issue, we can incorporate more semantic cues, *e.g.*, semantic labels, to make sure that important semantic objects should not be removed. This can be realized by either taking semantic information as an additional input or employing it as a mask or a confidence map to constrain the filtering process.

**(2) Task-driven smoothing**

Currently, image smoothing is performed on the entire image domain, *i.e.*, covering every pixel. It would be time-consuming when processing a high-resolution (large spatial size) image. Moreover, in many tasks, we are only interested in certain regions. For example, for object segmentation, we only care about the object and its

surroundings, so specially enhancing them will be more efficient. The regions can be selected either through semantic segmentation or manually.

**(3) Natural data for training and evaluation**

Although we present synthetic data in Chapter 4 for training the deep filtering model and evaluating other methods, they inevitably have domain gaps with natural images. Recently, Zhu et al. [2019b] have proposed a new dataset composed of natural images, but the ground truth images are from hand-crafted filters with manually-tuned parameters. In essence, training the network based on their data is still like approximating existing filters, which may be affected by their shortcomings in differentiating between structures and textures. In fact, it is quite difficult to label natural images for image smoothing because textures spread across the image and most of them do not have regular patterns and clear boundaries. Moreover, some textures are very tiny in scale, so annotating them is even harder. How to effectively label natural images is still under our consideration.

### 7.2.2 Depth Completion

**(1) Correcting LiDAR noise**

In the sparse depth input, there always exist obvious errors (referred to as *LiDAR noise*) around object boundaries, particularly with moving and small objects as well as partially transparent ones [Merriaux et al., 2017; Qiu et al., 2019]. A few studies have noted this noise problem and attempted to construct a confidence map to measure the reliability of LiDAR scans [Eldesokey, 2018; Van Gansbeke et al., 2019; Qiu et al., 2019]. For those points with potentially larger input errors, they lower the confidence to separate them from the interpolation process. However, this decreases the number of available depth points, making the input more sparse. Alternatively, we can correct these noisy points prior to feeding them into the completion network. A further problem here is that ground truth depth in real-world datasets is not purely dense, *i.e.*, not covering all positions. In that case, ground truth cannot supervise every valid pixel in the input, so some noisy points cannot be corrected. To handle this, we can take advantage of stereo fusion [Cheng et al., 2019] to generate fully dense pseudo labels for correcting input LiDAR noise in a data-driven manner. Stereo information is no longer required in testing. We think correcting LiDAR noise could further improve the depth completion performance.

**(2) Using other cues for guidance**

The RGB image can indeed provide more structural cues, but it lacks geometric information that is also necessary for depth completion. 3D information, *e.g.*, point clouds and surface normals, is more beneficial for reflecting the overall structure of the scene, reducing ambiguities around occluded boundaries, and distinguishing foreground/background objects. Existing studies [Xu et al., 2019; Chen et al., 2019] extract geometric features from pre-generated 3D and incorporate them into depth via a specially-designed feature fusion module. In our future work, we plan to obtain these features directly from the 2D depth map by reconstructing 3D from it (similar to image reconstruction in our work). This facilitates the acquisition of more depth-

related features and reduces the needs of additional 3D data at test time. To this end, we need to consider two main problems: (1) how to better extract 3D features from depth (and/or the image if it is combined with depth); and (2) how to bridge the gap between 2D and 3D features more effectively.

**(3) Enhancing the accuracy in distant regions**

The depth completion accuracy is always worse in distant regions than near ones (see Fig. 5.7(c)). This is caused by the perspective effect where distant objects are smaller and more densely distributed. In that case, object structures are less clear and CNN features are less reliable and informative. To enhance the accuracy, we can try two methods: (1) penalizing more on errors from distant regions (the MSE loss discussed in Chapter 5 somewhat has this effect but it is not specially aimed for distant points); and (2) designing a global structure guidance which gives a coarse estimation of overall depth (only roughly depicting near and far regions and temporarily ignoring detailed objects) and then refining it by recovering more details.

# Bibliography


Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; et al., 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, (2016). (cited on page 62)

Abdullah-Al-Wadud, M.; Kabir, M. H.; Dewan, M. A. A.; and Chae, O., 2007. A dynamic histogram equalization for image contrast enhancement. *IEEE Transactions on Consumer Electronics*, 53, 2 (2007), 593–600. (cited on page 1)

Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; and Süsstrunk, S., 2012. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34, 11 (2012), 2274–2282. (cited on page 15)

Adams, A.; Baek, J.; and Davis, M. A., 2010. Fast high-dimensional filtering using the permutohedral lattice. In *Computer Graphics Forum*, vol. 29, 753–762. Wiley Online Library. (cited on page 15)

Ahmad, S. A.; Taib, M. N.; Khalid, N. E. A.; and Taib, H., 2012. An analysis of image enhancement techniques for dental x-ray image interpretation. *International Journal of Machine Learning and Computing*, 2, 3 (2012), 292. (cited on page 1)

Alhaija, H.; Mustikovela, S.; Mescheder, L.; Geiger, A.; and Rother, C., 2018. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision (IJCV)*, (2018). (cited on page 86)

Ancuti, C. O. and Ancuti, C., 2013. Single image dehazing by multi-scale fusion. *IEEE Transactions on Image Processing*, 22, 8 (2013), 3271–3282. (cited on page 1)

Arbelaez, P.; Maire, M.; Fowlkes, C.; and Malik, J., 2010. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33, 5 (2010), 898–916. (cited on pages xix, xx, 64, 66, 67, 69, and 71)

Argyriou, A.; Evgeniou, T.; and Pontil, M., 2007. Multi-task feature learning. In *Advances in neural information processing systems*, 41–48. (cited on pages 75 and 77)

Arici, T.; Dikbas, S.; and Altunbasak, Y., 2009. A histogram modification framework and its application for image contrast enhancement. *IEEE Trans Image Process*, 18, 9 (2009), 1921–1935. (cited on page 1)

Atapour-Abarghouei, A. and Breckon, T. P., 2019. Veritatem dies aperit-temporally consistent depth prediction enabled by a multi-task geometric and semantic scene understanding approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3373–3384. (cited on page 75)

Aujol, J.-F.; Gilboa, G.; Chan, T.; and Osher, S., 2006. Structure-texture image decomposition—modeling, algorithms, and parameter selection. *International journal of computer vision*, 67, 1 (2006), 111–136. (cited on page 19)

Badrinarayanan, V.; Kendall, A.; and Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39, 12 (2017), 2481–2495. (cited on page 95)

Bae, S.; Paris, S.; and Durand, F., 2006. Two-scale tone management for photographic look. *ACM Transactions on Graphics (TOG)*, 25, 3 (2006), 637–645. (cited on page 14)

Bao; Paul; Zhang, L.; and Wu, X., 2005. Canny edge detection enhancement by scale multiplication. *IEEE Transactions on Pattern Analysis  Machine Intelligence*, (2005). (cited on page 1)

Bao, L.; Song, Y.; Yang, Q.; Yuan, H.; and Wang, G., 2014. Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree. *IEEE Transactions on Image Processing*, 23, 2 (2014), 555–569. (cited on page 15)

Barbero, A. and Sra, S., 2011. Fast newton-type methods for total variation regularization. ICML'11, 313–320. Omnipress. (cited on page 25)

Barron, J. T. and Poole, B., 2016. The fast bilateral solver. In *European Conference on Computer Vision*, 617–632. Springer. (cited on pages xxvi, 8, 20, 25, 90, 100, and 102)

Bengio, Y.; Lamblin, P.; Popovici, D.; and Larochelle, H., 2007. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*. (cited on page 92)

Burger, H. C.; Schuler, C. J.; and Harmeling, S., 2012. Image denoising: Can plain neural networks compete with bm3d? In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. (cited on page 1)

Cai, Y. and Baciu, G., 2012. Higher level segmentation: Detecting and grouping of invariant repetitive patterns. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 694–701. IEEE. (cited on page 15)

Cai, Y. and Baciu, G., 2013. Detecting, grouping, and structure inference for invariant repetitive patterns in images. *IEEE Transactions on Image Processing*, 22, 6 (2013), 2343–2355. (cited on page 57)

Canny, J., 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, , 6 (1986), 679–698. (cited on pages 16 and 20)

Chen, C.; Cai, J.; Zheng, J.; Cham, T.-J.; and Shi, G., 2013. A color-guided, region-adaptive and depth-selective unified framework for kinect depth recovery. In *2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*, 007–012. doi:10.1109/MMSP.2013.6659255. (cited on page 25)

Chen, J.; Adams, A.; Wadhwa, N.; and Hasinoff, S. W., 2016a. Bilateral guided upsampling. *ACM Transactions on Graphics (TOG)*, 35, 6 (2016), 1–8. (cited on page 24)

Chen, L.; Lin, H.; and Li, S., 2012. Depth image enhancement for kinect using region growing and bilateral filter. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 3070–3073. IEEE. (cited on page 7)

Chen, Q.; Xu, J.; and Koltun, V., 2017a. Fast image processing with fully-convolutional networks. In *The IEEE International Conference on Computer Vision (ICCV)*. (cited on pages xix, xxv, 5, 22, 54, 61, 62, 63, 64, and 65)

Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Fidler, S.; and Urtasun, R., 2016b. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2147–2156. (cited on page 7)

Chen, X.; Ma, H.; Wan, J.; Li, B.; and Xia, T., 2017b. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 1907–1915. (cited on page 25)

Chen, X. and Schmitt, F., 1992. Intrinsic surface properties from surface triangulation. In *European Conference on Computer Vision*, 739–743. Springer. (cited on page 34)

Chen, Y.; Yang, B.; Liang, M.; and Urtasun, R., 2019. Learning joint 2d-3d representations for depth completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10023–10032. (cited on pages 34, 84, and 109)

Chen, Z.; Badrinarayanan, V.; Drozdov, G.; and Rabinovich, A., 2018. Estimating depth from rgb and sparse sensing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 167–182. (cited on page 32)

Cheng, X.; Wang, P.; Guan, C.; and Yang, R., 2020. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 10615–10622. (cited on page 31)

Cheng, X.; Wang, P.; and Yang, R., 2018. Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 103–119. (cited on pages 8, 31, 74, 81, 84, 85, 87, and 90)

Cheng, X.; Zhong, Y.; Dai, Y.; Ji, P.; and Li, H., 2019. Noise-aware unsupervised deep lidar-stereo fusion. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 109)

Chien, C.-C.; Kinoshita, Y.; and Kiya, H., 2019. A noise-aware enhancement method for underexposed images. In *2019 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, 131–134. IEEE. (cited on pages xv and 2)

Cho, H.; Lee, H.; Kang, H.; and Lee, S., 2014. Bilateral texture filtering. *ACM Transactions on Graphics (TOG)*, 33, 4 (2014), 128. (cited on pages xvii, 16, and 42)

Cho, S. W.; Na, R. B.; Koo, J. H.; Arsalan, M.; and Kang, R. P., 2020. Semantic segmentation with low light images by modified cyclegan-based image enhancement. *IEEE Access*, PP, 99 (2020), 1–1. (cited on page 1)

Chodosh, N.; Wang, C.; and Lucey, S., 2018. Deep convolutional compressed sensing for lidar depth completion. In *Asian Conference on Computer Vision*, 499–513. Springer. (cited on pages 81 and 93)

Cimpoi, M.; Maji, S.; Kokkinos, I.; Mohamed, S.; ; and Vedaldi, A., 2014. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 56 and 58)

Coates, A.; Ng, A.; and Lee, H., 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 215–223. (cited on page 92)

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3213–3223. (cited on page 86)

Dai, L.; Yuan, M.; Zhang, F.; and Zhang, X., 2015. Fully connected guided image filtering. In *Proceedings of the IEEE International Conference on Computer Vision*, 352–360. (cited on page 15)

Dai, W.; Yang, Q.; Xue, G.-R.; and Yu, Y., 2007. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, 193–200. ACM. (cited on pages 59 and 77)

Dalal, N. and Triggs, B., 2005. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, 886–893. Ieee. (cited on page 23)

Dana, K. J.; Van Ginneken, B.; Nayar, S. K.; and Koenderink, J. J., 1999. Reflectance and texture of real-world surfaces. *ACM Transactions On Graphics (TOG)*, 18, 1 (1999), 1–34. (cited on pages 56 and 58)

Danielyan, A.; Katkovnik, V.; and Egiazarian, K., 2012. Bm3d frames and variational image deblurring. *IEEE Transactions on Image Processing*, 21, 4 (2012), 1715–1728. (cited on page 1)

Diebel, J. and Thrun, S., 2005. An application of markov random fields to range sensing. In *NIPS*, vol. 5, 291–298. (cited on page 25)

Dimitrievski, M.; Veelaert, P.; and Philips, W., 2018. Learning morphological operators for depth completion. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, 450–461. Springer. (cited on page 32)

Ding, Z.; Xu, Y.; Xu, W.; Parmar, G.; Yang, Y.; Welling, M.; and Tu, Z., 2020. Guided variational autoencoder for disentanglement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7920–7929. (cited on page 92)

Dollár, P. and Zitnick, C. L., 2015. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence*, 37, 8 (2015), 1558–1570. (cited on pages 16 and 20)

Dong, C.; Loy, C. C.; He, K.; and Tang, X., 2016. Image super-resolution using deep convolutional networks. *IEEE Trans Pattern Anal Mach Intell*, 38, 2 (2016), 295–307. (cited on page 1)

Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; and Koltun, V., 2017. Carla: An open urban driving simulator. In *Conference on robot learning*, 1–16. PMLR. (cited on page 33)

Dou, Z.; Song, M.; Gao, K.; and Jiang, Z., 2017. Image smoothing via truncated total variation. *IEEE Access*, 5 (2017), 27337–27344. (cited on page 19)

Durand, F. and Dorsey, J., 2002. Fast bilateral filtering for the display of high-dynamic-range images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 257–266. (cited on page 14)

Eisemann, E. and Durand, F., 2004. Flash photography enhancement via intrinsic relighting. *ACM transactions on graphics (TOG)*, 23, 3 (2004), 673–678. (cited on pages xvi, 16, and 17)

Eldesokey, A., 2018. Propagating confidences through cnns for sparse data regression. In *The British Machine Vision Conference (BMVC)*. (cited on pages 90 and 109)

Eldesokey, A.; Felsberg, M.; Holmquist, K.; and Persson, M., 2020. Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12014–12023. (cited on pages 30 and 32)

Eldesokey, A.; Felsberg, M.; and Khan, F. S., 2019. Confidence propagation through cnns for guided sparse depth regression. *IEEE transactions on pattern analysis and machine intelligence*, 42, 10 (2019), 2423–2436. (cited on pages xx, xxi, 7, 8, 27, 30, 74, 81, 82, 84, 85, 87, 88, and 93)

Fan, Q.; Wipf, D. P.; Hua, G.; and Chen, B., 2017a. Revisiting deep image smoothing and intrinsic image decomposition. *CoRR*, abs/1701.02965 (2017). http://arxiv.org/abs/1701.02965. (cited on pages 5 and 54)

Fan, Q.; Yang, J.; Hua, G.; Chen, B.; and Wipf, D., 2017b. A generic deep architecture for single image reflection removal and image smoothing. In *Proceedings of the IEEE International Conference on Computer Vision*, 3238–3247. (cited on pages xix, xxv, 3, 5, 22, 54, 61, 62, 63, 64, and 65)

Fan, Q.; Yang, J.; Wipf, D.; Chen, B.; and Tong, X., 2018. Image smoothing via unsupervised learning. *ACM Transactions on Graphics (TOG)*, 37, 6 (2018), 1–14. (cited on page 23)

Fang, S.; Yao, Z.; and Zhang, J., 2019a. Scale and gradient aware image smoothing. *IEEE Access*, 7 (2019), 166268–166281. (cited on pages 5 and 19)

Fang, X.; Wang, M.; Shamir, A.; and Hu, S.-M., 2019b. Learning explicit smoothing kernels for joint image filtering. In *Computer Graphics Forum*, vol. 38, 181–190. Wiley Online Library. (cited on page 23)

Farbman, Z.; Fattal, R.; Lischinski, D.; and Szeliski, R., 2008. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics (TOG)*, vol. 27, 67. ACM. (cited on pages xvii, 3, 19, 20, 40, and 41)

Fattal, R.; Agrawala, M.; and Rusinkiewicz, S., 2007. Multiscale shape and detail enhancement from multi-light image collections. *ACM Trans. Graph.*, 26, 3 (2007), 51. (cited on pages 3 and 40)

Ferstl, D.; Reinbacher, C.; Ranftl, R.; Rüther, M.; and Bischof, H., 2013. Image guided depth upsampling using anisotropic total generalized variation. In *Proceedings of the IEEE International Conference on Computer Vision*, 993–1000. (cited on pages xxvi, 8, 25, 85, 87, 90, 100, 102, 104, and 106)

Frieze, A. M., 1985. On the value of a random minimum spanning tree problem. *Discrete Applied Mathematics*, 10, 1 (1985), 47–56. (cited on page 15)

Fu, C.; Dong, C.; Mertz, C.; and Dolan, J. M., 2020. Depth completion via inductive fusion of planar lidar and monocular camera. *arXiv preprint arXiv:2009.01875*, (2020). (cited on pages 29 and 34)

Fukushima, N.; Sugimoto, K.; and Kamata, S.-i., 2018. Guided image filtering with arbitrary window function. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1523–1527. IEEE. (cited on page 17)

Gaidon, A.; Wang, Q.; Cabon, Y.; and Vig, E., 2016. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4340–4349. (cited on pages 33 and 93)

Galun, M.; Basri, R.; and Brandt, A., 2007. Multiscale edge detection and fiber enhancement using differences of oriented means. In *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*. (cited on page 1)

Gastal, E. S. and Oliveira, M. M., 2012. Adaptive manifolds for real-time high-dimensional filtering. *ACM Transactions on Graphics (TOG)*, 31, 4 (2012), 1–13. (cited on page 14)

George, E. B. and Karnan, M., 2012. Mri brain image enhancement using filtering techniques. *International Journal of Computer Science & Engineering Technology (IJCSET), ISSN*, (2012), 2229–3345. (cited on page 1)

Ghosh, S.; Gavaskar, R. G.; Panda, D.; and Chaudhury, K. N., 2019. Fast scale-adaptive bilateral texture smoothing. *IEEE Transactions on Circuits and Systems for Video Technology*, 30, 7 (2019), 2015–2026. (cited on page 14)

Gonzalez, R. C. and Woods, R. E., 1977. Digital image processing. *Addison-Wesley Pub. Co., Advanced Book Program*, (1977). (cited on pages 1 and 2)

Graham, B. and van der Maaten, L., 2017. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, (2017). (cited on page 26)

Gu, J.; Xiang, Z.; Ye, Y.; and Wang, L., 2021. Denselidar: A real-time pseudo dense depth guided depth completion network. *IEEE Robotics and Automation Letters*, 6, 2 (2021), 1808–1815. (cited on pages 30 and 32)

Gu, S.; Zhang, L.; Zuo, W.; and Feng, X., 2014. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2862–2869. (cited on pages 3 and 40)

Gu, S.; Zuo, W.; Guo, S.; Chen, Y.; Chen, C.; and Zhang, L., 2017. Learning dynamic guidance for depth image enhancement. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3769–3778. (cited on page 7)

Guo, X.; Li, S.; Li, L.; and Zhang, J., 2018. Structure-texture decomposition via joint structure discovery and texture smoothing. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. IEEE. (cited on page 20)

Guo, X.; Li, Y.; and Ma, J., 2017. Mutually guided image filtering. In *Proceedings of the 25th ACM international conference on Multimedia*, 1283–1290. (cited on page 21)

Guo, X.; Li, Y.; Ma, J.; and Ling, H., 2020. Mutually guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42, 3 (2020), 694–707. doi: 10.1109/TPAMI.2018.2883553. (cited on pages 2, 5, 16, and 21)

Hallman, S. and Fowlkes, C. C., 2015. Oriented edge forests for boundary detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1732–1740. (cited on pages xvii, 6, 41, and 42)

Ham, B.; Cho, M.; and Ponce, J., 2015. Robust image filtering using joint static and dynamic guidance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4823–4831. (cited on pages xviii, 20, 48, 49, 50, and 52)

Ham, B.; Cho, M.; and Ponce, J., 2017. Robust guided image filtering using nonconvex potentials. *IEEE transactions on pattern analysis and machine intelligence*, 40, 1 (2017), 192–207. (cited on pages xv, xviii, xix, xxv, 3, 4, 20, 54, 55, 62, 63, 64, and 65)

Haralick, R. M.; Shanmugam, K.; et al., 1973. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, , 6 (1973), 610–621. (cited on pages 15 and 24)

Hawe, S.; Kleinsteuber, M.; and Diepold, K., 2011. Dense disparity maps from sparse disparity measurements. In *2011 International Conference on Computer Vision*, 2126–2133. IEEE. (cited on page 25)

He, K. and Sun, J., 2012. Statistics of patch offsets for image completion. In *European conference on computer vision*, 16–29. Springer. (cited on page 3)

He, K. and Sun, J., 2014. Image completion approaches using the statistics of similar patches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36, 12 (2014), 2423–2435. (cited on page 1)

He, K.; Sun, J.; and Tang, X., 2010. Single image haze removal using dark channel prior. *IEEE transactions on pattern analysis and machine intelligence*, 33, 12 (2010), 2341–2353. (cited on page 1)

He, K.; Sun, J.; and Tang, X., 2013. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, 35, 6 (2013), 1397–1409. (cited on pages xv, xvi, xvii, xviii, xxv, 1, 3, 4, 14, 17, 22, 29, 40, 48, 49, 50, 52, 54, 55, 62, and 63)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37, 9 (2015), 1904–1916. (cited on page 28)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778. (cited on pages xvii, 22, 27, 28, 29, 32, and 57)

He, L. and Wang, Y., 2018. Image smoothing via truncated gradient regularisation. *IET Image Processing*, 12, 2 (2018), 226–234. (cited on page 19)

Henry, P.; Krainin, M.; Herbst, E.; Ren, X.; and Fox, D., 2012. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31, 5 (2012), 647–663. (cited on pages 8 and 74)

Hinton; G.; E.; Salakhutdinov; R.; and R., 2006. Reducing the dimensionality of data with neural networks. *Science*, (2006). (cited on pages 91, 92, and 94)

Hinton, G. E. and Zemel, R. S., 1993. Autoencoders, minimum description length and helmholtz free energy. *Advances in neural information processing systems*, 6 (1993). (cited on pages 91 and 94)

Hore, A. and Ziou, D., 2010. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, 2366–2369. IEEE. (cited on page 63)

Hornácek, M.; Rhemann, C.; Gelautz, M.; and Rother, C., 2013. Depth super resolution by rigid body self-similarity in 3d. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1123–1130. (cited on page 24)

Hosseini-Asl, E.; Zurada, J. M.; and Nasraoui, O., 2016. Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints. *IEEE Transactions on Neural Networks & Learning Systems*, 27, 12 (2016), 2486–2498. (cited on page 92)

Hu, C.; Wu, X.-J.; and Shu, Z.-Q., 2018. Discriminative feature learning via sparse autoencoders with label consistency constraints. *Neural Processing Letters*, (2018). (cited on page 92)

Hu, M.; Wang, S.; Li, B.; Ning, S.; Fan, L.; and Gong, X., 2021. Penet: Towards precise and efficient image guided depth completion. *arXiv preprint arXiv:2103.00783*, (2021). (cited on pages 29, 30, and 34)

Hua, K.-L.; Lo, K.-H.; and Wang, Y.-C. F. F., 2015. Extended guided filtering for depth map upsampling. *IEEE MultiMedia*, 23, 2 (2015), 72–83. (cited on page 24)

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q., 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708. (cited on page 32)

Huang, T. and Aizawa, K., 1993. Image processing: some challenging problems. *Proceedings of the National Academy of Sciences*, 90, 21 (1993), 9766–9769. (cited on page 2)

Huang, Z.; Fan, J.; Cheng, S.; Yi, S.; Wang, X.; and Li, H., 2019. Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. *IEEE Transactions on Image Processing*, 29 (2019), 3429–3441. (cited on pages 26 and 29)

Hussein, S. A.; Tirer, T.; and Giryes, R., 2020. Correction filter for single image super-resolution: Robustifying off-the-shelf deep super-resolvers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1428–1437. (cited on pages xv and 2)

Imran, S.; Liu, X.; and Morris, D., 2021. Depth completion with twin surface extrapolation at occlusion boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2583–2592. (cited on page 33)

Imran, S.; Long, Y.; Liu, X.; and Morris, D., 2019. Depth coefficients for depth completion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12438–12447. IEEE. (cited on page 33)

Jaritz, M.; De Charette, R.; Wirbel, E.; Perrotton, X.; and Nashashibi, F., 2018. Sparse and dense data with cnns: Depth completion and semantic segmentation. In *2018 International Conference on 3D Vision (3DV)*, 52–60. IEEE. (cited on pages 8, 26, 27, 28, 81, 84, and 90)

Jevnisek, R. J. and Avidan, S., 2017. Co-occurrence filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3184–3192. (cited on pages 5, 15, and 19)

Karacan, L.; Erdem, E.; and Erdem, A., 2013. Structure-preserving image smoothing via region covariances. *ACM Transactions on Graphics (TOG)*, 32, 6 (2013), 176. (cited on pages 5 and 14)

Katkovnik, V.; Egiazarian, K.; and Astola, J., 2005. A spatially adaptive nonparametric regression image deblurring. *IEEE Transactions on Image Processing*, 14, 10 (2005), 1469–1478. (cited on page 1)

Katkovnik, V.; Foi, A.; Egiazarian, K.; and Astola, J., 2010. From local kernel to nonlocal multiple-model image denoising. *International journal of computer vision*, 86, 1 (2010), 1. (cited on page 15)

Kendall, A. and Gal, Y., 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, 5574–5584. (cited on page 32)

Kendall, A.; Gal, Y.; and Cipolla, R., 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7482–7491. (cited on page 75)

Kerl, C.; Stuckler, J.; and Cremers, D., 2015. Dense continuous-time tracking and mapping with rolling shutter rgb-d cameras. In *Proceedings of the IEEE international conference on computer vision*, 2264–2272. (cited on pages 8 and 74)

Khetkeeree, S. and Thanakitivirul, P., 2020. Hybrid filtering for image sharpening and smoothing simultaneously. In *2020 35th International Technical Conference on*

*Circuits/Systems, Computers and Communications (ITC-CSCC)*, 367–371. IEEE. (cited on page 14)

Kim, B.; Ponce, J.; and Ham, B., 2021. Deformable kernel networks for joint image filtering. *International Journal of Computer Vision*, 129, 2 (2021), 579–600. (cited on page 23)

Kim, D. and Yoon, K.-j., 2012. High-quality depth map up-sampling robust to edge noise of range sensors. In *2012 19th IEEE International Conference on Image Processing*, 553–556. IEEE. (cited on page 25)

Kim, J.; Jeon, G.; and Jeong, J., 2014. Joint-adaptive bilateral depth map upsampling. *Signal Processing: Image Communication*, 29, 4 (2014), 506–513. (cited on page 24)

Kim, S.; Song, C.; Jang, J.; and Paik, J., 2019a. Edge-aware image filtering using a structure-guided cnn. *IET Image Processing*, 14, 3 (2019), 472–479. (cited on page 22)

Kim, Y.; Ham, B.; Do, M. N.; and Sohn, K., 2019b. Structure-texture image decomposition using deep variational priors. *IEEE Transactions on Image Processing*, 28, 6 (2019), 2692–2704. doi:10.1109/TIP.2018.2889531. (cited on page 5)

Kingma, D. P. and Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, (2014). (cited on pages 81 and 97)

Knutsson, H. and Westin, C.-F., 1993. Normalized and differential convolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 515–523. IEEE. (cited on page 27)

Köhler, R.; Schuler, C.; Schölkopf, B.; and Harmeling, S., 2014. Mask-specific inpainting with deep neural networks. In *German conference on pattern recognition*, 523–534. Springer. (cited on page 25)

Komodakis, N., 2006. Image completion using global optimization. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, 442–452. IEEE. (cited on page 3)

Komodakis, N. and Tziritas, G., 2007. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Transactions on Image Processing*, 16, 11 (2007), 2649–2661. (cited on page 1)

Kopf, J.; Cohen, M. F.; Lischinski, D.; and Uyttendaele, M., 2007. Joint bilateral upsampling. In *ACM Transactions on Graphics (ToG)*, vol. 26, 96. ACM. (cited on pages 2, 8, 16, and 90)

Kou, F.; Chen, W.; Wen, C.; and Li, Z., 2015. Gradient domain guided image filtering. *IEEE Transactions on Image Processing*, 24, 11 (2015), 4528–4539. (cited on page 17)

Krähenbühl, P. and Koltun, V., 2011. Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24 (2011), 109–117. (cited on page 20)

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25 (2012), 1097–1105. (cited on page 57)

Ku, J.; Harakeh, A.; and Waslander, S. L., 2018. In defense of classical image processing: Fast depth completion on the cpu. In *2018 15th Conference on Computer and Robot Vision (CRV)*, 16–22. IEEE. (cited on page 24)

Le, L.; Patterson, A.; and White, M., 2018. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. In *Advances in Neural Information Processing Systems*, 107–117. (cited on page 92)

Lee, B.-U.; Jeon, H.-G.; Im, S.; and Kweon, I. S., 2019a. Depth completion with deep geometry and context guidance. In *2019 International Conference on Robotics and Automation (ICRA)*, 3281–3287. IEEE. (cited on page 34)

Lee, B.-U.; Lee, K.; and Kweon, I. S., 2021. Depth completion using plane-residual representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13916–13925. (cited on page 32)

Lee, S.; Lee, J.; Kim, D.; and Kim, J., 2020. Deep architecture with cross guidance between single image and sparse lidar data for depth completion. *IEEE Access*, 8 (2020), 79801–79810. (cited on pages 29 and 30)

Lee, W.; Na, J.; and Kim, G., 2019b. Multi-task self-supervised object detection via recycling of bounding box annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4984–4993. (cited on page 75)

Li, A.; Yuan, Z.; Ling, Y.; Chi, W.; Zhang, C.; et al., 2020a. A multi-scale guided cascade hourglass network for depth completion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 32–40. (cited on page 29)

Li, J.; Lu, Z.; Zeng, G.; Gan, R.; and Zha, H., 2014a. Similarity-aware patchwork assembly for depth image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3374–3381. (cited on page 25)

Li, J.; Wang, N.; Zhang, L.; Du, B.; and Tao, D., 2020b. Recurrent feature reasoning for image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7760–7768. (cited on pages xv and 2)

Li, L.; Guo, X.; Feng, W.; and Zhang, J., 2018. Soft clustering guided image smoothing. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. IEEE. (cited on page 15)

Li, S. Z., 1994. Markov random field models in computer vision. In *European conference on computer vision*, 361–370. Springer. (cited on page 25)

Li, W.; Jafari, O. H.; and Rother, C., 2017. Semantic-aware image smoothing. In *Proceedings of the conference on Vision, Modeling and Visualization*, 153–160. (cited on page 20)

Li, Y.; Huang, J.-B.; Ahuja, N.; and Yang, M.-H., 2016. Deep joint image filtering. In *European Conference on Computer Vision*, 154–169. Springer. (cited on pages xvi, xxv, 5, 22, 23, 54, 61, 63, and 64)

Li, Y.; Huang, J.-B.; Ahuja, N.; and Yang, M.-H., 2019. Joint image filtering with deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 41, 8 (2019), 1909–1923. (cited on page 22)

Li, Z. and Ji, X., 2020. Pose-guided auto-encoder and feature-based refinement for 6-dof object pose regression. In *IEEE International Conference on Robotics and Automation (ICRA), 2020*. (cited on page 92)

Li, Z. and Zheng, J., 2017. Single image de-hazing using globally guided image filtering. *IEEE Transactions on Image Processing*, 27, 1 (2017), 442–450. (cited on page 3)

Li, Z.; Zheng, J.; and Zhu, Z., 2014b. Content adaptive guided image filtering. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. IEEE. (cited on page 17)

Li, Z.; Zheng, J.; Zhu, Z.; Yao, W.; and Wu, S., 2014c. Weighted guided image filtering. *IEEE Transactions on Image processing*, 24, 1 (2014), 120–129. (cited on page 17)

Liang, M.; Yang, B.; Chen, Y.; Hu, R.; and Urtasun, R., 2019. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7345–7353. (cited on page 75)

Liao, Y.; Huang, L.; Wang, Y.; Kodagoda, S.; Yu, Y.; and Liu, Y., 2017. Parse geometry from a line: Monocular depth estimation with partial laser observation. In *2017 IEEE international conference on robotics and automation (ICRA)*, 5059–5066. IEEE. (cited on page 32)

Liebel, L. and Körner, M., 2018. Auxiliary tasks in multi-task learning. *arXiv preprint arXiv:1805.06334*, (2018). (cited on page 75)

Liu, J. and Gong, X., 2013. Guided depth enhancement via anisotropic diffusion. In *Pacific-Rim conference on multimedia*, 408–417. Springer. (cited on page 24)

Liu, J.; Zhang, W.; Tang, Y.; Tang, J.; and Wu, G., 2020a. Residual feature aggregation network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2359–2368. (cited on pages xv and 2)

Liu, L.; Liao, Y.; Wang, Y.; Geiger, A.; and Liu, Y., 2021. Learning steering kernels for guided depth completion. *IEEE Transactions on Image Processing*, 30 (2021), 2850–2861. (cited on pages 32 and 104)

Liu, L.; Song, X.; Lyu, X.; Diao, J.; Wang, M.; Liu, Y.; and Zhang, L., 2020b. Fcfr-net: Feature fusion based coarse-to-fine residual learning for monocular depth completion. *arXiv preprint arXiv:2012.08270*, (2020). (cited on page 32)

Liu, M.-Y.; Tuzel, O.; and Taguchi, Y., 2013a. Joint geodesic upsampling of depth images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 169–176. (cited on page 24)

Liu, N.; Zhang, N.; and Han, J., 2020c. Learning selective self-mutual attention for rgb-d saliency detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13756–13765. (cited on page 28)

Liu, Q.; Liu, J.; Dong, P.; and Liang, D., 2013b. Sgtd: Structure gradient and texture decorrelating regularization for image decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, 1081–1088. (cited on pages xxv, 19, 41, 43, 48, 55, 62, and 63)

Liu, S.; De Mello, S.; Gu, J.; Zhong, G.; Yang, M.-H.; and Kautz, J., 2017a. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/c22abfa379f38b5b0411bc11fa9bf92f-Paper.pdf. (cited on page 31)

Liu, S.; Pan, J.; and Yang, M.-H., 2016. Learning recursive filters for low-level vision via a hybrid neural network. In *European Conference on Computer Vision*, 560–576. Springer. (cited on pages xxv, 5, 22, 54, 63, and 64)

Liu, S.; Wang, Y.; Wang, J.; Wang, H.; Zhang, J.; and Pan, C., 2013c. Kinect depth restoration via energy minimization with tv21 regularization. In *2013 IEEE International Conference on Image Processing*, 724–724. doi:10.1109/ICIP.2013.6738149. (cited on page 25)

Liu, W.; Chen, X.; Shen, C.; Liu, Z.; and Yang, J., 2017b. Semi-global weighted least squares in image filtering. In *Proceedings of the IEEE International Conference on Computer Vision*, 5861–5869. (cited on page 19)

Liu, W.; Chen, X.; Shen, C.; Yu, J.; Wu, Q.; and Yang, J., 2017c. Robust guided image filtering. *arXiv preprint arXiv:1703.09379*, (2017). (cited on page 17)

Liu, W.; Zhang, P.; Chen, X.; Shen, C.; Huang, X.; and Yang, J., 2018a. Embedding bilateral filter in least squares for efficient edge-preserving image smoothing. *IEEE Transactions on Circuits and Systems for Video Technology*, 30, 1 (2018), 23–35. (cited on page 20)

Liu, W.; Zhang, P.; Huang, X.; Yang, J.; Shen, C.; and Reid, I., 2020d. Real-time image smoothing via iterative least squares. *ACM Transactions on Graphics (TOG)*, 39, 3 (2020), 1–24. (cited on page 18)

Liu, X.; Zhai, D.; Bai, Y.; Ji, X.; and Gao, W., 2019. Contrast enhancement via dual graph total variation-based image decomposition. *IEEE Transactions on Circuits and Systems for Video Technology*, 30, 8 (2019), 2463–2476. (cited on pages xv, 2, and 3)

Liu, Y.; Jourabloo, A.; and Liu, X., 2018b. Learning deep models for face anti-spoofing: Binary or auxiliary supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 389–398. (cited on page 75)

Liu, Y.; Ma, X.; Li, X.; and Zhang, C., 2020e. Two-stage image smoothing based on edge-patch histogram equalisation and patch decomposition. *IET image processing*, 14, 6 (2020), 1132–1140. (cited on pages 5 and 20)

Lo, K.-H.; Wang, Y.-C. F.; and Hua, K.-L., 2017. Edge-preserving depth map upsampling by joint trilateral filter. *IEEE transactions on cybernetics*, 48, 1 (2017), 371–384. (cited on page 16)

Lopez-Rodriguez, A.; Busam, B.; and Mikolajczyk, K., 2020. Project to adapt: Domain adaptation for depth completion from noisy and sparse sensor data. In *Proceedings of the Asian Conference on Computer Vision*. (cited on page 33)

Lu, J.; Shi, K.; Min, D.; Lin, L.; and Do, M. N., 2012. Cross-based local multipoint filtering. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 430–437. IEEE. (cited on page 15)

Lu, K.; Barnes, N.; Anwar, S.; and Zheng, L., 2020. From depth what can you see? depth completion via auxiliary image reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11306–11315. (cited on pages xxii, xxvi, 9, 11, 90, 96, 97, 98, 99, 103, 104, and 106)

Lu, K.; Barnes, N.; Anwar, S.; and Zheng, L., 2022. Depth completion auto-encoder. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, 63–73. (cited on pages 10 and 11)

Lu, K.; You, S.; and Barnes, N., 2017. Double-guided filtering: Image smoothing with structure and texture guidance. In *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 1–8. IEEE. (cited on pages xix, xxv, 6, 10, 54, 56, 62, 63, 64, 65, and 66)

Lu, K.; You, S.; and Barnes, N., 2018a. Deep texture and structure aware filtering network for image smoothing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 217–233. (cited on pages xv, 2, 7, and 10)

Lu, S.; Ren, X.; and Liu, F., 2014. Depth enhancement via low-rank matrix completion. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 3390–3397. doi:10.1109/CVPR.2014.433. (cited on page 24)

Lu, X.; Guo, Y.; Liu, N.; Wan, L.; and Fang, T., 2018b. Non-convex joint bilateral guided depth upsampling. *Multimedia Tools and Applications*, 77, 12 (2018), 15521–15544. (cited on page 24)

Lv, F.; Li, Y.; and Lu, F., 2021. Attention guided low-light image enhancement with a large scale low-light simulation dataset. *International Journal of Computer Vision*, (2021), 1–19. (cited on page 3)

Ma, C.; Rao, Y.; Cheng, Y.; Chen, C.; Lu, J.; and Zhou, J., 2020. Structure-preserving super resolution with gradient guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7769–7778. (cited on page 3)

Ma, F.; Cavalheiro, G. V.; and Karaman, S., 2018. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. *arXiv preprint arXiv:1807.00275*, (2018). (cited on pages 81 and 103)

Ma, F.; Cavalheiro, G. V.; and Karaman, S., 2019. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In *2019 International Conference on Robotics and Automation (ICRA)*, 3288–3295. IEEE. (cited on pages xv, xvi, xvii, xx, xxi, xxii, xxvi, 7, 8, 9, 27, 28, 29, 35, 74, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 96, 98, 99, and 100)

Ma, F. and Karaman, S., 2018. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 4796–4803. IEEE. (cited on pages 28, 85, 86, and 87)

Masci, J.; Angulo, J.; and Schmidhuber, J., 2013. A learning framework for morphological operators using counter–harmonic mean. In *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*, 329–340. Springer. (cited on page 28)

Masci, J.; Meier, U.; Dan, C.; and Schmidhuber, J., 2011. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*. (cited on page 92)

Merriaux, P.; Dupuis, Y.; Boutteau, R.; Vasseur, P.; and Savatier, X., 2017. Lidar point clouds correction acquired from a moving car based on can-bus data. *arXiv preprint arXiv:1706.05886*, (2017). (cited on page 109)

Miao, D.; Fu, J.; Lu, Y.; Li, S.; and Chen, C. W., 2012. Texture-assisted kinect depth inpainting. In *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, 604–607. doi:10.1109/ISCAS.2012.6272103. (cited on page 24)

Min, D.; Choi, S.; Lu, J.; Ham, B.; Sohn, K.; and Do, M. N., 2014. Fast global image smoothing based on weighted least squares. *IEEE Transactions on Image Processing*, 23, 12 (2014), 5638–5653. (cited on page 19)

Mun, J.; Jang, Y.; and Kim, J., 2018. Propagated guided image filtering for edge-preserving smoothing. *Signal, Image and Video Processing*, 12, 6 (2018), 1165–1172. (cited on page 17)

Newell, A.; Yang, K.; and Deng, J., 2016. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, 483–499. Springer. (cited on page 29)

Nguyen, R. M. and Brown, M. S., 2015. Fast and effective l0 gradient minimization by region fusion. In *Proceedings of the IEEE International Conference on Computer Vision*, 208–216. (cited on pages xviii, xxv, 19, 48, 49, 50, 51, 52, 54, 55, 62, and 63)

Ni, K. and Wu, Y., 2018. Adaptive patched l0 gradient minimisation model applied on image smoothing. *IET Image Processing*, 12, 10 (2018), 1892–1902. (cited on page 19)

Noroozi, M.; Vinjimoor, A.; Favaro, P.; and Pirsiavash, H., 2018. Boosting self-supervised learning via knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9359–9367. (cited on page 59)

Pan, J.; Dong, J.; Ren, J. S.; Lin, L.; Tang, J.; and Yang, M.-H., 2019. Spatially variant linear representation models for joint filtering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1702–1711. (cited on page 18)

Paris, S. and Durand, F., 2006. A fast approximation of the bilateral filter using a signal processing approach. In *European conference on computer vision*, 568–580. Springer. (cited on page 14)

Park, J.; Joo, K.; Hu, Z.; Liu, C.-K.; and Kweon, I. S., 2020. Non-local spatial propagation network for depth completion. In *European Conference on Computer Vision (ECCV)*. (cited on pages 30 and 31)

Park, J.; Kim, H.; Tai, Y.-W.; Brown, M. S.; and Kweon, I., 2011. High quality depth map upsampling for 3d-tof cameras. In *2011 International Conference on Computer Vision*, 1623–1630. doi:10.1109/ICCV.2011.6126423. (cited on page 25)

Park, J.; Kim, H.; Tai, Y.-W.; Brown, M. S.; and Kweon, I. S., 2014. High-quality depth map upsampling and completion for rgb-d cameras. *IEEE Transactions on Image Processing*, 23, 12 (2014), 5559–5572. doi:10.1109/TIP.2014.2361034. (cited on page 25)

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A., 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*. (cited on pages 81 and 97)

Perona, P. and Malik, J., 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12, 7 (1990), 629–639. (cited on page 20)

Petschnigg, G.; Szeliski, R.; Agrawala, M.; Cohen, M.; Hoppe, H.; and Toyama, K., 2004. Digital photography with flash and no-flash image pairs. *ACM transactions on graphics (TOG)*, 23, 3 (2004), 664–672. (cited on pages xvi, 2, 16, and 17)

Pham, Q.-H.; Nguyen, T.; Hua, B.-S.; Roig, G.; and Yeung, S.-K., 2019. Jsis3d: Joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8827–8836. (cited on page 75)

Polesel, A.; Ramponi, G.; and Mathews, V. J., 2000. Image enhancement via adaptive unsharp masking. *IEEE transactions on image processing*, 9, 3 (2000), 505–510. (cited on page 23)

Porikli, F., 2008. Constant time o (1) bilateral filtering. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. IEEE. (cited on page 14)

Qi, X.; Liao, R.; Liu, Z.; Urtasun, R.; and Jia, J., 2018. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 283–291. (cited on page 34)

Qiu, J.; Cui, Z.; Zhang, Y.; Zhang, X.; Liu, S.; Zeng, B.; and Pollefeys, M., 2019. Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (cited on pages xxi, 7, 8, 28, 30, 32, 33, 34, 74, 81, 85, 86, 87, 90, 106, and 109)

Qu, C.; Nguyen, T.; and Taylor, C., 2020. Depth completion via deep basis fitting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 71–80. (cited on pages 28 and 104)

Ranjan, A.; Jampani, V.; Balles, L.; Kim, K.; Sun, D.; Wulff, J.; and Black, M. J., 2019. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 7)

Riemens, A.; Gangwal, O.; Barenbrug, B.; and Berretty, R.-P., 2009. Multistep joint bilateral depth upsampling. In *Visual communications and image processing 2009*, vol. 7257, 72570M. International Society for Optics and Photonics. (cited on page 24)

Romera, E.; Alvarez, J. M.; Bergasa, L. M.; and Arroyo, R., 2017. Efficient convnet for real-time semantic segmentation. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, 1789–1794. IEEE. (cited on page 29)

Romera-Paredes, B.; Argyriou, A.; Berthouze, N.; and Pontil, M., 2012. Exploiting unrelated tasks in multi-task learning. In *International conference on artificial intelligence and statistics*, 951–959. (cited on pages 75 and 77)

Rong, C.; Qu, Y.; Li, C.; Zeng, K.; and Li, C., 2018. Single-image super-resolution via joint statistic models-guided deep auto-encoder network. *Neural Computing & Applications*, 32, 2 (2018), 1–12. (cited on page 92)

Ronneberger, O.; Fischer, P.; and Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer. (cited on page 32)

Rudin, L. I.; Osher, S.; and Fatemi, E., 1992. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60, 1-4 (1992), 259–268. (cited on pages xviii, xxv, 4, 18, 48, 49, 62, and 63)

Schneider, N.; Schneider, L.; Pinggera, P.; Franke, U.; Pollefeys, M.; and Stiller, C., 2016. Semantically guided depth upsampling. In *German conference on pattern recognition*, 37–48. Springer. (cited on pages 8 and 25)

Schuster, R.; Wasenmuller, O.; Unger, C.; and Stricker, D., 2021. Ssgp: Sparse spatial guided propagation for robust and generic interpolation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 197–206. (cited on page 29)

Shang, R.; Liu, M.; Lin, J.; Feng, J.; Li, Y.; Stolkin, R.; and Jiao, L., 2021. Sar image segmentation based on constrained smoothing and hierarchical label correction. *IEEE Transactions on Geoscience and Remote Sensing*, (2021). (cited on pages xvi, 13, 14, and 15)

Sharma, V.; Hardeberg, J. Y.; and George, S., 2017. Rgb-nir image enhancement by fusing bilateral and weighted least squares filters. *Journal of Imaging Science and Technology*, 61, 4 (2017), 40409–1. (cited on page 16)

Shen, C.-T.; Chang, F.-J.; Hung, Y.-P.; and Pei, S.-C., 2012. Edge-preserving image decomposition using l1 fidelity with l0 gradient. In *SIGGRAPH Asia 2012 Technical Briefs*, 6. ACM. (cited on page 19)

Shen, X.; Chen, Y.; Tao, X.; and Jia, J., 2017. Convolutional neural pyramid for image processing. *CoRR*, abs/1704.02071 (2017). http://arxiv.org/abs/1704.02071. (cited on pages 5 and 54)

Shen, X.; Zhou, C.; Xu, L.; and Jia, J., 2015. Mutual-structure for joint filtering. In *Proceedings of the IEEE International Conference on Computer Vision*, 3406–3414. (cited on page 20)

Shi, F.; Cheng, J.; Wang, L.; Yap, P.-T.; and Shen, D., 2015. Lrtv: Mr image super-resolution with low-rank and total variation regularizations. *IEEE transactions on medical imaging*, 34, 12 (2015), 2459–2466. (cited on page 3)

Shi, Z.; Chen, Y.; Gavves, E.; Mettes, P.; and Snoek, C. G., 2021. Unsharp mask guided filtering. *arXiv preprint arXiv:2106.01428*, (2021). (cited on page 23)

Shivakumar, S. S.; Nguyen, T.; Miller, I. D.; Chen, S. W.; Kumar, V.; and Taylor, C. J., 2019. Dfusenet: Deep fusion of rgb and sparse depth information for image guided dense depth completion. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 13–20. IEEE. (cited on page 28)

Shukla, K. N.; Potnis, A.; and Dwivedy, P., 2017. A review on image enhancement techniques. *International Journal of Engineering and Applied Computer Science (IJEACS)*, 2, 7 (2017), 232–235. (cited on page 1)

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R., 2012. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, 746–760. Springer. (cited on pages xxvi, 8, 24, 73, 85, 86, 87, 89, 90, 100, 102, 104, and 106)

Snoek, J.; Adams, R. P.; and Larochelle, H., 2012. Nonparametric guidance of autoencoder representations using label information. *Journal of Machine Learning Research*, 13, 1 (2012), 2567–2588. (cited on page 92)

Starck, J.-L.; Moudden, Y.; Bobin, J.; Elad, M.; and Donoho, D., 2005. Morphological component analysis. In *Wavelets XI*, vol. 5914, 59140Q. International Society for Optics and Photonics. (cited on page 19)

Subr, K.; Soler, C.; and Durand, F., 2009. Edge-preserving multiscale image decomposition based on local extrema. *ACM Transactions on Graphics (TOG)*, 28, 5 (2009), 147. (cited on pages xvi, 5, 18, and 19)

Sun, Z.; Han, B.; Li, J.; Zhang, J.; and Gao, X., 2019. Weighted guided image filtering with steering kernel. *IEEE Transactions on Image Processing*, 29 (2019), 500–508. (cited on page 17)

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A., 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9. (cited on page 77)

Tan, X.; Sun, C.; and Pham, T. D., 2014. Multipoint filtering with local polynomial approximation and range guidance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2941–2948. (cited on page 15)

Tang, J.; Tian, F.-P.; Feng, W.; Li, J.; and Tan, P., 2020. Learning guided convolutional network for depth completion. *IEEE Transactions on Image Processing*, 30 (2020), 1116–1129. (cited on page 29)

Tao, C.; Ma, K. K.; and Chen, L. H., 2000. Tri-state median filter for image denoising. *IEEE Transactions on Image Processing*, 8, 12 (2000), 1834–1838. (cited on page 1)

Teng, Y.; Liu, Y.; Yang, J.; Li, C.; Qi, S.; Kang, Y.; Fan, F.; and Wang, G., 2019. Graph regularized sparse autoencoders with nonnegativity constraints. *Neural Processing Letters*, 50, 1 (2019), 247–262. (cited on page 92)

Tomasi, C. and Manduchi, R., 1998. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, 839–846. IEEE. (cited on pages xvii, xviii, xxv, 1, 3, 4, 5, 14, 20, 40, 44, 48, 49, 50, 52, 62, and 63)

Tung, T.-C. and Fuh, C.-S., 2021. Icebin: Image contrast enhancement based on induced norm and local patch approaches. *IEEE Access*, 9 (2021), 23737–23750. (cited on page 3)

Tuzel, O.; Porikli, F.; and Meer, P., 2006. Region covariance: A fast descriptor for detection and classification. In *European conference on computer vision*, 589–600. Springer. (cited on page 14)

Uhrig, J.; Schneider, N.; Schneider, L.; Franke, U.; Brox, T.; and Geiger, A., 2017. Sparsity invariant cnns. In *2017 International Conference on 3D Vision (3DV)*, 11–20. IEEE. (cited on pages xv, xvi, 1, 7, 8, 11, 25, 26, 27, 73, 74, 75, 81, 87, 89, 90, 93, 97, 104, and 106)

Valada, A.; Radwan, N.; and Burgard, W., 2018. Deep auxiliary learning for visual localization and odometry. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 6939–6946. IEEE. (cited on page 76)

Van Gansbeke, W.; Neven, D.; De Brabandere, B.; and Van Gool, L., 2019. Sparse and noisy lidar completion with rgb guidance and uncertainty. In *2019 16th international conference on machine vision applications (MVA)*, 1–6. IEEE. (cited on pages xx, xxi, 8, 30, 32, 74, 81, 82, 84, 85, 88, 90, and 109)

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P.-A., 2008. Extracting and composing robust features with denoising autoencoders. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*. (cited on page 92)

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P. A., 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11, 12 (2010), 3371–3408. (cited on pages 92 and 101)

Vogels, T.; Rousselle, F.; McWilliams, B.; Röthlin, G.; Harvill, A.; Adler, D.; Meyer, M.; and Novák, J., 2018. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics (TOG)*, 37, 4 (2018), 1–15. (cited on page 33)

Wang, A.; Xu, Y.; Wei, X.; and Cui, B., 2020. Semantic segmentation of crop and weed using an encoder-decoder network and image enhancement method under uncontrolled outdoor illumination. *IEEE Access*, PP, 99 (2020), 1–1. (cited on page 1)

Wang, D. C.; Vagnucci, A. H.; and Li, C.-C., 1983. Digital image enhancement: a survey. *Computer Vision, Graphics, and Image Processing*, 24, 3 (1983), 363–381. (cited on page 1)

Wang, S. and Ding, Z., 2017. Feature selection guided auto-encoder. In *AAAI*. (cited on page 92)

Wang, S.; Ding, Z.; and Fu, Y., 2016. Coupled marginalized auto-encoders for cross-domain multi-view learning. In *IJCAI*, 2125–2131. (cited on page 92)

Wang, S.; Suo, S.; Ma, W.-C.; Pokrovsky, A.; and Urtasun, R., 2018. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2589–2597. (cited on page 34)

Wang, Y.; Chao, W.-L.; Garg, D.; Hariharan, B.; Campbell, M.; and Weinberger, K. Q., 2019a. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 7)

Wang, Y. and He, C., 2012. Image segmentation algorithm by piecewise smooth approximation. *EURASIP Journal on Image and Video Processing*, 2012, 1 (2012), 16. (cited on page 3)

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M., 2019b. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38, 5 (2019), 1–12. (cited on page 31)

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P., 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13, 4 (2004), 600–612. (cited on pages 36 and 63)

Winnemöller, H.; Olsen, S. C.; and Gooch, B., 2006. Real-time video abstraction. In *ACM Transactions On Graphics (TOG)*, vol. 25, 1221–1226. ACM. (cited on pages 3, 40, and 51)

Wong, A.; Cicek, S.; and Soatto, S., 2021a. Learning topology from synthetic data for unsupervised depth completion. *IEEE Robotics and Automation Letters*, 6, 2 (2021), 1495–1502. (cited on pages xxii, xxvi, 9, 34, 35, 36, 91, 92, 93, 96, 98, 99, and 100)

Wong, A.; Fei, X.; Hong, B.-W.; and Soatto, S., 2021b. An adaptive framework for learning unsupervised depth completion. *IEEE Robotics and Automation Letters*, 6, 2 (2021), 3120–3127. (cited on pages 35 and 36)

Wong, A.; Fei, X.; Tsuei, S.; and Soatto, S., 2020. Unsupervised depth completion from visual inertial odometry. *IEEE Robotics and Automation Letters*, 5, 2 (2020), 1899–1906. (cited on pages xxii, xxvi, 9, 35, 36, 91, 92, 93, 96, 98, 99, and 100)

Wu, H.; Zheng, S.; Zhang, J.; and Huang, K., 2018. Fast end-to-end trainable guided filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1838–1847. (cited on page 22)

Xie, J.; Feris, R. S.; and Sun, M.-T., 2015. Edge-guided single depth image super resolution. *IEEE Transactions on Image Processing*, 25, 1 (2015), 428–438. (cited on page 25)

Xie, S. and Tu, Z., 2015. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, 1395–1403. (cited on pages 6, 20, 56, 61, and 66)

Xu, L.; Lu, C.; Xu, Y.; and Jia, J., 2011. Image smoothing via l 0 gradient minimization. In *ACM Transactions on Graphics (TOG)*, vol. 30, 174. ACM. (cited on pages xvii, xviii, xxv, 3, 4, 19, 40, 41, 50, 51, 52, 62, and 63)

Xu, L.; Ren, J.; Yan, Q.; Liao, R.; and Jia, J., 2015. Deep edge-aware filters. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 1669–1678. (cited on pages xvi, xxv, 5, 21, 54, 61, 63, and 64)

Xu, L.; Yan, Q.; Xia, Y.; and Jia, J., 2012. Structure extraction from texture via relative total variation. *ACM transactions on graphics (TOG)*, 31, 6 (2012), 1–10. (cited on pages xvii, xviii, xix, xxv, 19, 42, 48, 49, 54, 58, 59, 60, 62, and 63)

Xu, R.; Xu, Y.; and Quan, Y., 2021. Structure-texture image decomposition using discriminative patch recurrence. *IEEE Transactions on Image Processing*, 30 (2021), 1542–1555. doi:10.1109/TIP.2020.3043665. (cited on page 19)

Xu, Y.; Zhu, X.; Shi, J.; Zhang, G.; Bao, H.; and Li, H., 2019. Depth completion from sparse lidar data with depth-normal constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2811–2820. (cited on pages xxi, 30, 34, 74, 81, 84, 86, 87, and 109)

Xu, Z.; Yin, H.; and Yao, J., 2020. Deformable spatial propagation networks for depth completion. In *2020 IEEE International Conference on Image Processing (ICIP)*, 913–917. IEEE. (cited on pages 30 and 31)

Yalniz, I. Z. and Aksoy, S., 2010. Unsupervised detection and localization of structural textures using projection profiles. *Pattern Recognition*, 43, 10 (2010), 3324–3337. (cited on page 57)

Yan, L.; Liu, K.; and Belyaev, E., 2020. Revisiting sparsity invariant convolution: A network for image guided depth completion. *IEEE Access*, 8 (2020), 126323–126332. (cited on page 29)

Yan, Q.; Shen, X.; Xu, L.; Zhuo, S.; Zhang, X.; Shen, L.; and Jia, J., 2013. Cross-field joint image restoration via scale map. In *Proceedings of the IEEE International Conference on Computer Vision*, 1537–1544. (cited on pages 2 and 16)

Yang, J.; Qi, Z.; and Shi, Y., 2020. Learning to incorporate structure knowledge for image inpainting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 12605–12612. (cited on page 3)

Yang, J.; Ye, X.; Li, K.; Hou, C.; and Wang, Y., 2014. Color-guided depth recovery from rgb-d data using an adaptive autoregressive model. *IEEE Transactions on Image Processing*, 23, 8 (2014), 3443–3458. doi:10.1109/TIP.2014.2329776. (cited on page 24)

Yang, J. T. Y., J.Wright, 2010. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19, 11 (2010), 2861–2873. (cited on page 1)

Yang, Q., 2015. Recursive approximation of the bilateral filter. *IEEE transactions on image processing*, 24, 6 (2015), 1919–1927. (cited on page 14)

Yang, Q., 2016. Semantic filtering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4517–4526. (cited on pages 16 and 48)

Yang, Q.; Ahuja, N.; and Tan, K.-H., 2015. Constant time median and bilateral filtering. *International Journal of Computer Vision*, 112, 3 (2015), 307–318. (cited on page 14)

Yang, Q.; Tan, K.-H.; and Ahuja, N., 2009. Real-time o (1) bilateral filtering. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 557–564. IEEE. (cited on page 14)

Yang, Y. and Soatto, S., 2018. Conditional prior networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 271–287. (cited on pages 33 and 36)

Yang, Y.; Wong, A.; and Soatto, S., 2019. Dense depth posterior (ddp) from single image and sparse range. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3353–3362. (cited on pages xvi, xxvi, 9, 33, 35, 36, 91, 92, 93, 96, 98, and 99)

Yao, Y.; Roxas, M.; Ishikawa, R.; Ando, S.; Shimamura, J.; and Oishi, T., 2020. Discontinuous and smooth depth completion with binary anisotropic diffusion tensor. *IEEE Robotics and Automation Letters*, 5, 4 (2020), 5128–5135. (cited on page 24)

Ye, J.; Ji, Y.; Wang, X.; Ou, K.; Tao, D.; and Song, M., 2019. Student becoming the master: Knowledge amalgamation for joint scene parsing, depth estimation, and more. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 7)

Yin, W.; Goldfarb, D.; and Osher, S., 2005. Image cartoon-texture decomposition and feature selection using the total variation regularized l 1 functional. In *International Workshop on Variational, Geometric, and Level Set Methods in Computer Vision*, 73–84. Springer. (cited on page 19)

Yu, F. and Koltun, V., 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, (2015). (cited on page 22)

Yu, W.; Zeng, G.; Luo, P.; Zhuang, F.; He, Q.; and Shi, Z., 2013. Embedding with autoencoder regularization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 208–223. Springer. (cited on page 92)

Zang, Y.; Huang, H.; and Zhang, L., 2015. Guided adaptive image smoothing via directional anisotropic structure measurement. *IEEE transactions on visualization and computer graphics*, 21, 9 (2015), 1015–1027. (cited on page 16)

Zeng, K.; Yu, J.; Wang, R.; Li, C.; and Tao, D., 2015. Coupled deep autoencoder for single image super-resolution. *IEEE Transactions on Cybernetics*, (2015), 27–37. (cited on page 92)

Zhang, C.; Tang, Y.; Zhao, C.; Sun, Q.; Ye, Z.; and Kurths, J., 2021. Multitask gans for semantic segmentation and depth completion with cycle consistency. *IEEE Transactions on Neural Networks and Learning Systems*, (2021). (cited on page 34)

Zhang, F.; Dai, L.; Xiang, S.; and Zhang, X., 2015. Segment graph based image filtering: Fast structure-preserving smoothing. In *Proceedings of the IEEE International Conference on Computer Vision*, 361–369. (cited on pages xviii, xxv, 4, 15, 40, 48, 49, 50, 52, 54, 55, 62, and 63)

Zhang, J.; Schroeder, J.; and Redding, N. J., 2003. Sar image enhancement for small target detection. In *IEEE International Conference on Acoustics*. (cited on page 1)

Zhang, J.; Tian, G.; Mu, Y.; and Fan, W., 2014a. Supervised deep learning with auxiliary networks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 353–361. ACM. (cited on page 75)

Zhang, K.; Gao, X.; Tao, D.; and Li, X., 2012. Single image super-resolution with non-local means and steering kernel regression. *IEEE Transactions on Image Processing*, 21, 11 (2012), 4544–4556. (cited on page 3)

Zhang, Q.; Shen, X.; Xu, L.; and Jia, J., 2014b. Rolling guidance filter. In *European Conference on Computer Vision*, 815–830. Springer. (cited on pages xviii, xxv, 16, 20, 48, 49, 50, 51, 52, 54, 55, 62, and 63)

Zhang, X.; Zhou, X.; Lin, M.; and Sun, J., 2018. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6848–6856. (cited on page 32)

Zhang, Y. and Funkhouser, T., 2018. Deep depth completion of a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 175–185. (cited on pages 34, 85, and 87)

Zhang, Y.; Nguyen, T.; Miller, I. D.; Shivakumar, S. S.; Chen, S.; Taylor, C. J.; and Kumar, V., 2019a. Dfinenet: Ego-motion estimation and depth refinement from sparse, noisy depth input with rgb guidance. *arXiv preprint arXiv:1903.06397*, (2019). (cited on pages 9 and 35)

Zhang, Z.; Cui, Z.; Xu, C.; Yan, Y.; Sebe, N.; and Yang, J., 2019b. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4106–4115. (cited on pages 7 and 75)

Zhao, H.; Gallo, O.; Frosio, I.; and Kautz, J., 2016. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3, 1 (2016), 47–57. (cited on page 61)

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J., 2017. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–2890. (cited on page 86)

Zhao, L.; Bai, H.; Liang, J.; Wang, A.; Zeng, B.; and Zhao, Y., 2019. Local activity-driven structural-preserving filtering for noise removal and image smoothing. *Signal Processing*, 157 (2019), 62–72. (cited on page 19)

Zhao, S.; Gong, M.; Fu, H.; and Tao, D., 2021. Adaptive context-aware multi-modal network for depth completion. *IEEE Transactions on Image Processing*, (2021). (cited on page 31)

Zhou, F.; Chen, Q.; Liu, B.; and Qiu, G., 2019. Structure and texture-aware image decomposition via training a neural network. *IEEE Transactions on Image Processing*, 29 (2019), 3458–3473. (cited on page 23)

Zhou, L. and Gu, X., 2018. Deep neural network based salient object detection with image enhancement. In *International Conference on Neural Information Processing*. (cited on page 1)

Zhu, A. Z.; Yuan, L.; Chaney, K.; and Daniilidis, K., 2019a. Unsupervised event-based learning of optical flow, depth, and egomotion. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 7)

Zhu, F.; Fang, C.; and Ma, K.-K., 2020. Pnen: Pyramid non-local enhanced networks. *IEEE Transactions on Image Processing*, 29 (2020), 8831–8841. (cited on page 22)

Zhu, F.; Liang, Z.; Jia, X.; Zhang, L.; and Yu, Y., 2019b. A benchmark for edge-preserving image smoothing. *IEEE Transactions on Image Processing*, 28, 7 (2019), 3556–3570. (cited on pages 22 and 109)

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2223–2232. (cited on page 33)

Zhu, L.; Hu, X.; Fu, C.-W.; Qin, J.; and Heng, P.-A., 2018. Saliency-aware texture smoothing. *IEEE transactions on visualization and computer graphics*, 26, 7 (2018), 2471–2484. (cited on page 20)

Zoph, B.; Vasudevan, V.; Shlens, J.; and Le, Q. V., 2018. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8697–8710. (cited on pages 26 and 28)