ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN

COMPUTER SCIENCE AND ENGINEERING

Ciclo __36___

Settore Concorsuale: __09/H1__

Settore Scientifico Disciplinare:___ING-INF/05___

Compact and Effective Models for Depth Prediction

Presentata da: ___*RIZHAO FAN*____

Coordinatore Dottorato
ILARIA BARTOLINI

Supervisore
STEFANO MATTOCCIA

Co-Supervisore
MATTEO POGGI

Esame finale anno 2024

To those who have been by my side.

Abstract

Depth prediction is at the core of several computer vision applications, such as autonomous driving, augmented reality, and robotics. Approaches for depth prediction can be divided into two main classes: active and passive sensing. Active depth sensing is the de-facto standard of applications requiring depth sensing on its excellent accuracy and low latency in varied environments. Passive depth sensing using cameras requires a large baseline and careful calibration to obtain accurate depth results. Deep learning has significantly facilitated the development of dense depth prediction, affecting the accuracy of models inferring depth from images or multi-modal data. Moreover, despite the wide literature concerning depth prediction, there are open problems. Most works adopted computationally expensive models, posing a significant challenge for devices with limited computational resources. Furthermore, current models seldom study the inherent characteristics of depth, which still hold significant potential.

This thesis focuses on addressing some issues related to depth prediction. Compact and effective models were proposed to recover dense results across diverse settings, both supervised and self-supervised methods, from color camera images and sparse LiDAR measurements. Additionally, by analyzing the characteristics of the depth map, contrastive learning techniques are introduced to improve the depth

prediction network's learning ability and unlock further potential. All experiments are validated on the commonly used datasets, including KITTI and the NYU depth v2 dataset, following the standard metrics to compare our proposals with previous representative state-of-the-art works.

Acknowledgements

I would like to acknowledge my supervisors Prof. Stefano Mattoccia and Prof. Matteo Poggi for their invaluable supervision, support and patience during my PhD. My gratitude extends to my teammates Fabio Tosi, Filippo Aleotti, Youmi Zhang, Huan Li with whom I enjoyed years of productive research and invaluable collaboration. Their contributions were invaluable.

I thank Prof. Antonio Manuel López and Dr. Gabriel Villalonga for allowing me to join and work with their team for three months at the Computer Vision Center (CVC), Barcelona. I learned a lot from this great experience thanks to them and each member of CVC.

I express profound gratitude to the China Scholarship Council for their generous funding and support, which made my three-year educational journey possible.

Lastly, my appreciation also goes out to my loved friends for their support throughout my studies. I extend heartfelt and deep appreciation to all my family members. Their love and support have always guided me. Thanks and love to my parents Xiaosu and Sijun. My brothers Rijie, Rike, and sister Jiao-Jiao, have consistently been by my side, offering encouragement and support as I pursued my dreams in a foreign land.

Contents

List of Figures

List of Tables

Introduction

Computer vision enables machines to interpret and understand visual information from images or videos, and contains tasks like object detection, image recognition, and 3D scene understanding, widely applied in fields like robotics and image analysis. Depth prediction is an important computer vision task that aims at measuring the distance of pixels in the image from the camera. Depth sensing methods can be categorized into two primary categories: active and passive sensing. Representatives of the active depth techniques are Laser Imaging Detection and Ranging (LiDAR), structured-light, and time-of-flight (ToF) cameras. They uniformly sample the depth of the entire scene by measuring the signal's travel time from emitter to receiver at a constant scan rate, providing high fidelity and precise depth information. Nonetheless, LiDARs are prohibitively expensive and provide only sparse point clouds. Structured-light and ToF cameras, such as the Microsoft Kinect, have limited range and are sensitive to ambient light interference. Passive depth sensing utilizes multi-view vision, inferring depth from environmental cues without emitting signals. It usually refers to the problem of reconstructing an accurate 3D scene structure from multiple images with known camera poses and intrinsic, prevalent in computer vision, robotics, and 3D imaging. Passive methods are typically less affected by adverse weather and offer advantages in simplicity and reduced power consumption.

The acquisition of dense and accurate depth maps is of utmost importance for various applications, such as autonomous driving, 3D scene reconstruction, and augmented reality. Existing active depth sensors, including LiDARs, structured-light-based depth sensors, and ToF cameras, fail to get dense depth maps and provide only sparse measurements due to inherent limitations. Passive depth sensing, relying on image disparities from multiple cameras, can generate dense depth maps. However, this approach depends on accurate triangulation, which is computational-consuming and usually fails at textureless regions or occlusions within complex scenes. These limitations have driven significant interest in inferring high-quality dense depth maps through cost-effective and energy-efficient approaches.

Traditional computer vision relies on handcrafted algorithms and predefined rules to interpret visual data. However, the advance of deep learning has revolutionized the field, showing its immense capabilities across various computer vision tasks. Deep learning-based computer vision has achieved remarkable accuracy in tasks such as image classification, object detection, depth prediction, and image segmentation. This technology also greatly promotes the advancement of depth prediction. Utilizing neural networks, depth prediction models infer high-quality 3D depth details from images or sparse point clouds [1, 2]. Driven by these successes, in this thesis, we will inquire about the application of deep learning to depth prediction. We propose compact and efficient depth prediction models from images and sparse accurate measurements, as well as study the features of depth distribution to enhance the models' learning. More specifically, The main topics covered by this thesis will be supervised compact depth completion network (Chapter 3) , together with a self-supervised lightweight depth estimation network with few LiDAR measurements (Chapter 4). In order to exploit more of the properties of the depth distribution, we will also introduce contrastive learning to improve the prediction model's learning without increased parameters and computational complexity (Chapter 5).

CHAPTER 1

Related Work

In this chapter, a thorough review of the main works relevant to this thesis will be reported. To understand this research thesis, this Chapter provides an extensive overview of related works in the fields of depth prediction, with a particular focus on depth completion, and monocular depth estimation. Moreover, we also review relevant literature about feature fusion, contrastive learning, and self-attention, and how these technologies have been adopted for this research thesis.

Depth completion. Depth completion aims at recovering dense depth maps from sparse and incomplete measurements. For accurate depth information sampling, high-performance LiDAR sensors are suitable for outdoor scenarios and Time-of-Flight (ToF) sensors, such as Microsoft Kinect, work well for indoor scenes. Hawe *et al.* [3] reconstructed disparity maps from very few measurements with a conjugate subgradient method. Liu *et al.* [4] proposed a combined wavelet-contourlet dictionary to estimate dense depth maps. Ma *et al.* [5] leveraged the regularity (e.g., many planar surfaces with few edges) in the depth to recover dense depth. Ku et al. [6] utilized a series of hand-crafted classical image processing algorithms to infer a dense depth map from image and sparse depth map inputs.

Learning methods greatly facilitate depth completion. Uhrig *et al.* [7] proposed a sparsity-invariant convolution layer to consider the location of missing data while addressing data sparsity within deep networks. Huang *et al.* [8] proposed sparsity-invariant multi-scale encoder-decoder network for sparse inputs and feature maps is also proposed. Ma *et al.* [9, 10] utilize early fusion to combine sparse depth with a color image and feed them into an encoder-decoder CNN, which boosts the performance of depth completion. Multi-branch network architecture is an effective approach to fuse multi-modal data as reported in [11, 12, 13, 14, 15]. Spatial propagation networks (SPN) [16] is another popular depth-refinement approach [17, 18, 19, 20, 21]. DeepLiDAR [13] introduces pixel-wise surface normals as geometric constraints and proposes multiple branches to generate dense depth maps jointly. GuideFormer [22] and CompletionFormer [23] introduce transformers in this task as well. Graph representations have been used [24, 25, 26] for better modeling the relationships between sparse point clouds, while transformers [22, 23] have been deployed to model long-range relationships. Recurrent networks can effectively

recover dense predictions from sparse representations [17, 18, 19, 20, 21, 27].

Depth estimation. Single image depth estimation is a computer vision task estimating the 3D structure or scene geometry from a single 2D image without the need for multiple images or stereo pairs [28]. It is crucial for various applications, including Augmented Reality, Autonomous Driving, and 3D Scene Reconstruction.

Early works on depth estimation using RGB images usually relied on hand-crafted features and probabilistic graphical models. Saxena *et al.* [29] estimated the absolute scales of different image patches and inferred the depth image using a Markov Random Field model. Karsch *et al.* [30, 31] proposed a technique for estimating depth from video by employing non-parametric sampling methods. Konrad *et al.* [32] exploited to estimate the depth of a query image by combining the depths of images with photometric content most closely matches retrieved from a database.

Regarding depth estimation, learning-based depth estimation approaches gained much interest in the literature in recent years. Eigen *et al.* [33, 34] proposed a multi-stage, coarse-to-fine network to estimate depth from a single image, and Liu *et al.* [35] formulated depth estimation into deep CNN and a continuous conditional random field, and attained visually sharper transitions and local details. Laina *et al.* [36] a fully convolutional architecture based on the ResNet for depth estimation. Some works introduce attention mechanisms to achieve significant performance improvements [37, 38, 39, 40, 41, 42, 43, 44]. There have been numerous mainstream methods formulating depth estimation by discretizing continuous depth range into discrete bins, and classification-and-regression problems [45, 46, 47, 48]. A high-order geometric constraint is also employed to reconstruct depth prediction in [49, 50]. Some multi-task works predict depth maps by jointly learning with other vision tasks [51, 52, 53, 54]. DANet [45] proposes to utilize depth distribution as supervision for the prediction.

Self-supervised depth estimation. In recent years, self-supervised monocular depth prediction has gained significant attention, with two primary training methods being explored using either stereo images [55] or monocular videos [56] in the absence of dense labels. Garg *et al.* [55] propose the first framework that uses an image reconstruction loss on stereo images to train a monocular depth model. In

contrast, Zhou *et al.* [56] leverage a framework that jointly estimates depth and pose by utilizing video sequences and a photometric loss at training time. Godard *et al.* [57] proposed an automatic occlusion method, Monodepth2, which minimized photometric error to reduce the artifacts at the object boundary, and improved the sharpness of the occlusion boundary. Subsequent works followed both paths [58, 59, 60, 61, 62, 63, 64], significantly improving the accuracy of self-supervised solutions and shrinking the gap with supervised ones. To address the moving object problem, some works also introduce an efficient strategy by introducing an additional loss to ignore dynamic objects [57, 65, 66, 67, 68].

Self-supervised depth estimation with sparse measurements. A very recent trend consists of estimating dense depth from images and sparse measurements, e.g., LiDAR data, in a self-supervised manner, reducing deployment costs at the minimum. We position this task at the intersection between self-supervised depth estimation and completion, given the minimal impact of the few LiDAR scans available with respect to the usual standard 64-beam setup for outdoor depth completion. Some of them [10, 69, 70, 71] construct such depth estimation network by minimizing the photometric error across monocular sequences, as well as minimizing the discrepancy between the sparse inputs and the dense outputs. Ma *et al.* [10] proposed a self-supervised training framework on sequences of color and sparse depth images with pose estimation using the PnP method. Choi *et al.* [71] designed a self-supervised network leveraging sparsity-invariant CNNs [7] to extract sparse depth features and pixel-adaptive convolutions to fuse image and depth features for challenging indoor environments. LidarStereoNet [72] proposed a Lidar-stereo fusion network in an unsupervised learning scheme. Feng *et al.* [73] proposed a representative solution in this field using a two-stage network to infer dense depth maps. LidarTouch [74] explored self-supervised depth estimation with few LiDAR data in multiple depth completion networks and pose estimation methods.

Lightweight Dense Prediction. There is practical demand for lightweight networks as more mobile and on-edge devices emerge. MobileNet [75, 76] and ShuffleNet [77], were developed specifically for devices with limited computing power. BiSeNet [78] and BiSeNetV2 [79] are lightweight networks for semantic segmenta-

tion, using two-stream paths for modeling low-level details and high level semantic information. PyD-Net [62] and PyD-Net2 [63] are pyramidal architectures for self-supervised monocular depth estimation, deployed on edge devices as well [80, 81, 82]. ICNet [83] uses cascade down-sampled images as input and fuses multi-scale features to pursue efficiency.

Feature-level Fusion approaches. Deep learning methods for depth completion usually aggregate depth and image information at the feature level. Lee et al. [84] propose a cross-guidance between image and depth encoder branches and fuse multi-modal features through attention. GuideNet [85] adopts image features as guidance and fuses multi-modal features with skip connections across encoder-decoder networks. FusionNet [14] adopts global branches to guide local branches by concatenating features from different branches.

Multi-level feature fusion also proved effective [86, 87]. Feature pyramid networks [88] utilizes a top-down architecture with lateral connections and fuse multi-scale feature through features sum. UNet++ [89] proposes a nested UNet to learn the importance of features at different layers and adopts a dense skip connection to aggregate multi-scale features. DFANet [90] develops a cross-level feature aggregation strategy to boost accuracy.

Contrastive learning. Contrastive learning has achieved remarkable progress employing discriminative learning by contrasting positive pairs against negative pairs in representations space [91] and some works target visual representation learning [92, 93, 94, 95, 96]. SimCLR [96] implements contrastive learning in a simple network framework, where positive pairs are from data augmentation of the same image, while negative ones are from different images. MoCo [94] maintains a queue of negative samples and turns one branch of a Siamese network into a momentum encoder to improve the queue consistency. Some recent works [97, 98, 99] have introduced contrastive learning for dense prediction. ReSim [100] learns regional representations from a pair of views originating by sliding windows from the same image. DenseCL [98] optimizes a pairwise contrastive loss at the pixel level between two different image views. Ke *et al.* [101] propose a weakly supervised segmentation method that utilizes contrastive relationships between pixels and segments in the

feature space. Alonso *et al.* [102] utilize a memory bank to contrast the features from labeled and unlabeled data employing end-to-end training. Some works [103, 104] use a contrastive loss to generate high-frequency details for image super-resolution tasks. Shen *et al.* [105] propose contrastive differential learning in image translation and use it for depth-to-depth synthesis.

Window-based Approaches. Long-range dependency is a notable cue and Transformers [106], Markov Random Fields (MRFs) [107] and Conditional Random Fields (CRFs) [108, 109] use it to boost their learning ability. However, these methods have a severe drawback in the computational complexity, increasing quadratically with image size. Purposely, some works aim at addressing this issue. Dosovitskiy *et al.* [110] apply a transformer on sequences of image patches for image classification tasks. Pyramid ViT [111] uses a progressive shrinking pyramid and spatial-reduction attention to reduce computations of the transformer on large feature maps. Swin Transformer [112] proposes a novel architecture that computes attention within a patch-based window and uses a shifting window approach to capture attention in non-overlapping regions. Yuan *et al.* [42] employ a window-based CRFs approach for monocular depth estimation. CSWin [113] proposes a self-attention within a cross-shaped window in different directions, which yields strong representation learning with limited computation cost.

Self-attention and Transformers. Self-attention is well-known for modeling long-range dependencies in learning. Wang *et al.* [114] introduced self-attention to computer vision and presented a novel non-local network with great success in multiple vision tasks. CCNet [115] further proposed sparsely-connected graphs to generate sparse attention maps through a criss-cross path, which can reduce the complexity.

Transformer networks in Natural Language Processing (NLP) have achieved great success, which sparked great interest in the computer vision community. Dosovitskiy *et al.* [110] is the first to apply pure Transformer architecture for visual recognition tasks and propose Vision Transformer (ViT), achieving astonishing performance on visual classification benchmarks. Wang *et al.* [111] proposed Pyramid ViT (PVT), a hierarchical design for ViT, and proposes a progressive shrinking pyramid and spatial-

reduction attention for various pixel-level dense prediction tasks. SegFormer [116] further improved the vision transformer's performance by introducing overlapping patch embedding, depth-wise convolution, and efficient attention. Swin Transformer [112] is a hierarchical Transformer architecture whose representation is computed with shifted windows.

CHAPTER 2

Datasets and Metrics

Contents

In this chapter, we introduce popular datasets and protocols used to evaluate predicted results for depth prediction tasks.

2.1 Reference datasets

Comparing recently introduced methods to prior research is feasible due to the availability of popular datasets providing dense ground-truth depth maps. In this chapter, we present two popular depth prediction datasets and their commonly used splits representing the most adopted datasets to measure the accuracy of depth prediction algorithms.

2.1.1 KITTI dataset.

KITTI benchmark [117] is an outdoor dataset captured from autonomous driving platforms for use in mobile robotics and autonomous driving, as shown in Fig 2.1. It is a popular and challenging real-world computer vision benchmark, including multiple tasks of interest: stereo, depth, optical flow, visual odometry, 3D object detection, and 3D tracking. In this thesis, we focus on depth prediction tasks.

Fig. 2.1: **KITTI Recording Platform.**

KITTI depth completion dataset[1, 7] is a popular outdoor dataset providing sparse depth maps captured by Velodyne LiDAR HDL-64e, color stereo images, and corresponding semi-dense ground truth, as shown in Fig. 2.2. In the depth completion task, the sparse depth maps provide 5.9% valid depth values on all pixels, while the ground truth maps contain 16% valid depth values over the whole image. The

dataset contains 85895 training frames, with 1000 more selected validation frames, and 1000 test data for which ground truth is withheld.

Fig. 2.2: **Samples from the KITTI depth completion dataset.** From top to bottom: RGB images, raw LiDAR datas, and annotated depth maps, respectively.

KITTI Eigen Split [33] is a popular data split, a subset of the full KITTI depth prediction dataset. When using this split, preprocessing was performed to remove static frames, as in [56, 57], thus, 39,910 and 4,424 images were used for training and validation, respectively, and 697 images were used for evaluation.

2.1.2 NYU depthv2 dataset.

The NYU Depth v2 dataset [2] is an indoor dataset with depth measurements acquired by a Microsoft Kinect device. It consists of 120K RGB images and depth maps at 480×640 resolution collected from 464 different indoor scenes captured by a Microsoft Kinect sensor, as shown in Fig. 2.3. In the depth completion task, for the training data, we utilized a subset of ∼50K images from the official training split. Each image was downsized to 320×240, and then 304×228 center-cropping was applied. The official test split of 654 images was used for evaluation and comparisons. In the depth estimation task, we used 249 scenes for training and 215 scenes (654 images) for testing, resulting in 24231 image depth pairs for the training set.

Fig. 2.3: **Samples of the RGB image, and the raw depth image from the NYU Depth v2 dataset.**

2.2 Evaluation Metrics.

In this chapter, we will report the standard protocols used to evaluate and compare the approaches proposed in this work of thesis with the state-of-the-art.

In KITTI depth completion task, we adopt the official evaluation protocol from the KITTI depth completion benchmark [7] to evaluate our work, computing four standard metrics: the mean absolute error (MAE, mm), root mean squared error (RMSE, mm), mean absolute error of the inverse depth (iMAE, 1/km) and root mean squared error of the inverse depth (iRMSE, 1/km). Among them, RMSE and MAE directly measure depth accuracy, while RMSE is more sensitive and selected to rank all the submitted methods on the KITTI leaderboard.

In NYU Depth Dataset v2 [2] depth prediction task, we use common metrics in the field: RMSE, Mean Absolute Relative Error (REL), Absolute relative difference (Abs Rel), Square relative difference (Sq Rel), log rmse (rmse log), average log error (Log10), and percentage of predicted pixels where the relative error is within a threshold (δ_i).

Among these evaluation metrics, RMSE measures the square root of the average of the squared differences between prediction and ground truth (GT).

$$RMSE = \sqrt{\frac{1}{V}\sum_{v \in V} \left| \, d_v^{gt} - d_v^{pred} \, \right|^2} \tag{2.1}$$

iRMSE calculates the square root of the average of the squared differences between the prediction and GT inverse depth values, typically expressed in units of 1/km.

$$iRMSE = \sqrt{\frac{1}{V}\sum_{v \in V} \left| \, 1/d_v^{gt} - 1/d_v^{pred} \, \right|^2} \tag{2.2}$$

RMSE and iRMSE both provide a measure of the overall accuracy by considering the differences between predictions and GT. RMSE focuses on depth values directly, while iRMSE focuses on inverse depth values, providing insights into the model's performance in different representations of depth.

MAE measures the average absolute difference between the prediction and GT. MAE provides a straightforward indication of the average magnitude of errors in depth prediction, without considering the direction of the errors.

$$MAE = \frac{1}{V}\sum_{v \in V} \left| \, d_v^{gt} - d_v^{pred} \, \right| \tag{2.3}$$

iMAE measures the mean absolute difference between the prediction and GT inverse depth values, usually expressed in units of 1/km.

$$iMAE = \frac{1}{V}\sum_{v \in V} \left| \, 1/d_v^{gt} - 1/d_v^{pred} \, \right| \tag{2.4}$$

Similar to RMSE and iRMSE, MAE and iMAE offer a measure of the average discrepancy between predictions and GT, but with an emphasis on absolute differences rather than squared differences. This can be useful for understanding the typical magnitude of errors without the influence of squared terms.

REL computes the mean of the absolute relative differences between prediction and ground truth (GT), normalized by GT.

$$REL = \frac{1}{V} \sum_{v \in V} \left| \ (d_v^{gt} - d_v^{pred})/d_v^{gt} \ \right| \tag{2.5}$$

Abs Rel indicates the absolute difference between prediction and GT, normalized by GT.

$$Abs\_Rel = \frac{1}{V} \sum_{v \in V} \frac{|d_v^{pred} - d_v^{gt}|}{d_v^{gt}} \tag{2.6}$$

REL and Abs Rel provide measures of relative errors, which normalize the errors by GT. They are useful for understanding the proportional accuracy of the predictions across different depth ranges. REL is particularly informative when considering errors relative to the magnitude of GT.

Sq Rel computes the squared difference between prediction and GT, normalized by GT.

$$Sq\_Rel = \frac{1}{V} \sum_{v \in V} \frac{(d_v^{pred} - d_v^{gt})^2}{d_v^{gt}} \tag{2.7}$$

RMSE Log measures the square root of the average of the squared differences between the logarithms of prediction and GT.

$$rmse\_log = \sqrt{\frac{1}{V} \sum_{v \in V} (log(d_v^{pred}) - log(d_v^{gt}))^2} \tag{2.8}$$

Log10 measures the mean of the absolute differences between the logarithms of prediction and GT, usually on base 10 logarithms.

$$Log\_10 = \frac{1}{V} \sum_{v \in V} |log(d_v^{pred}) - log(d_v^{gt})| \tag{2.9}$$

RMSE Log and Log10 metrics analyze the logarithms of depth values, offering insights into the model's performance on a logarithmic scale. They excel in capturing errors across a broad spectrum of depth values, addressing situations where linear metrics like RMSE might lack clarity in interpretation.

The percentage of predicted pixels where the relative error is within a threshold,

$$\delta_\tau = \texttt{max}\left(\frac{d_v^{gt}}{d_v^{pred}}, \frac{d_v^{pred}}{d_v^{gt}}\right) < \tau, \tau = 1.25, 1.25^2, 1.25^3 \tag{2.10}$$

CHAPTER 3

Compact and Effective Supervised Depth Completion

Contents

The content of this chapter has been presented at the 33rd British Machine Vision Conference (BMVC 2022) - "A Cascade Dense Connection Fusion Network for Depth Completion" [11].

3.1 Introduction

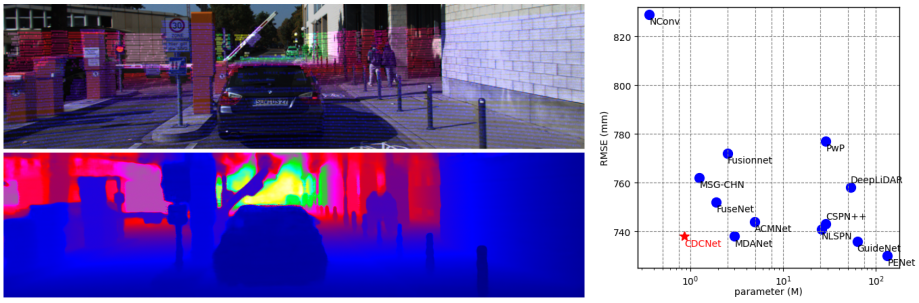


Fig. 3.1: **Cascade Dense Connection Fusion Network in action.** Our model predicts accurate dense depth maps from RGB frame and LiDAR points, using a fraction of the parameters compared to most of the existing methods.

With the benefit of a high-resolution color image and deep learning, current methods [14, 118, 119, 120] based on convolutional neural networks (CNNs) have made significant progress in inferring dense depth map from multi-modal data. Nevertheless, most of these existing depth completion methods rely on complex and heavy CNNs, unsuitable for in-vehicle and edge devices. Moreover, these models often use naive aggregation approaches, such as features concatenation or sum, resulting in sub-optimal strategies when fusing multi-modal data.

To tackle these problems, we propose a Cascade Dense Connection fusion network composed of a cascade of Dense Connection Fusion (DCF) blocks. Inspired by [12, 89, 121, 122], we stack our lightweight DCF blocks in a progressive manner instead of building a heavy encoder-decoder network, which allows for saving many parameters. More specifically, the DCF block can learn multi-modal and multi-level features by dense connections and multi-scale learning. We construct a Modality-Aware Aggregation module for learning the multi-modal representations and a Multi-Scale Pyramid Fusion module for learning multi-level features. Figure 3.1 plots the relationship between parameters and the primary evaluation metric, i.e., RMSE, for the proposed model and state-of-the-art depth completion approaches. We can

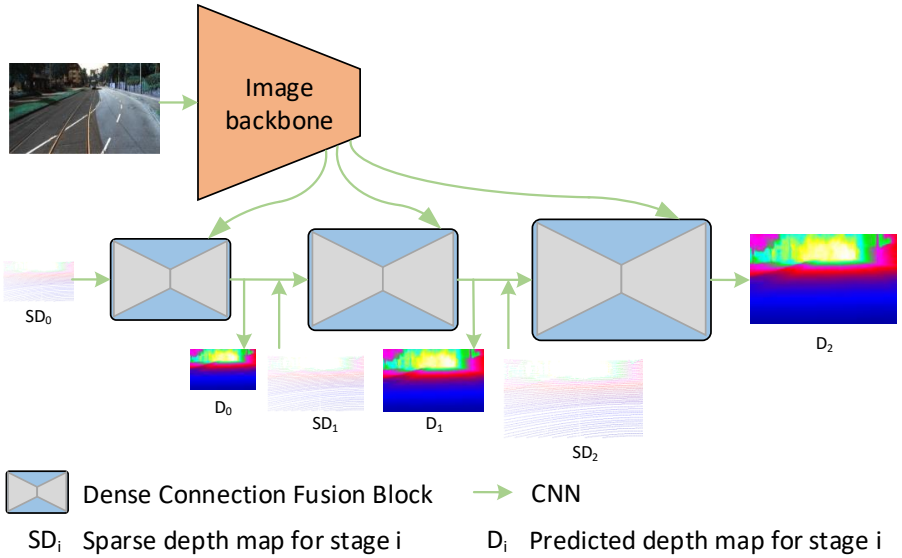


Fig. 3.2: **Pipeline of the proposed method.** The image backbone extracts image features and feeds them to dense connection fusion (DCF) blocks. Three DCF blocks are stacked progressively for three stages. DCF_0, DCF_1, DCF_2, from small to big.

notice how CDCNet achieves a favorable trade-off compared to existing methods.

We propose a lightweight Cascade Dense Connection fusion Network (CDCNet) for depth completion, which depends on dense connections to extract and learn depth and RGB features efficiently and effectively. We design Modality-Aware Aggregation (MAA) and Multi-Scale Pyramid Fusion (MSPF) modules for learning multi-modal and multi-level representations more effectively. And more, experimental results show that CDCNet is competitive with state-of-the-art approaches on the KITTI depth completion benchmark while counting much fewer parameters.

3.2 Cascade Dense Connection Fusion Network for Depth Completion

This section describes our proposal for effective and efficient depth completion performed by processing an RGB image and sparse depth data.

We first present our Cascade Dense Connection fusion Network. Then, in subsequent sections, we provide details for the proposed functional modules, i.e., Dense Connection Fusion block, Modality-Aware Aggregation module and Multi-Scale Pyra-

mid Fusion module. The overall architecture of CDCNet is depicted in Figure 3.2. It takes a color image and a sparse depth map to progressively recover a dense depth map. The image backbone consists of 10 convolutional layers with 3×3 filters. The 3th, 5th, 7th and 9th layers have stride 2 while the others have stride 1. The depth backbones are built using 6 layers defined following the same structure of the first 6 layers of the image backbone. All convolutions are followed by BatchNorm and ReLU functions. One image backbone and one depth backbone have 84K and 46K parameters, respectively. Following [12, 14, 15], we stack lightweight blocks instead of designing a heavy backbone to learn feature representations. The image backbone extracts multi-scale features, providing meaningful information on semantics and texture as guidance to recover depth. We denote the extracted image features as F_I^1, F_I^2, F_I^3, F_I^4, F_I^5, with cumulative strides of 1, 2, 4, 8, 16, respectively. Image features are then fed to the three cascade DCF blocks, namely DCF_0, DCF_1, DCF_2. Apart from image features, quarter-sized sparse map SD_0, half-sized depth SD_1, and full-sized depth map SD_2 are fed to the abovementioned three blocks. Each stage outputs a dense prediction at the same input size, with residual connections integrating the three outputs. Note that all the feature maps in our model have the same number of channels, i.e., $C = 32$, except for multi-scale learning parts. This way, our network is very lightweight. For simplicity, we omit some connection lines about residual connections in Figure 3.2. More details about how residual connections are integrated into each module can be found in [12].

3.2.1 Dense Connection Fusion Block

Most depth completion methods use naive concatenation or sum operations to aggregate either heterogeneous depth and image features or homogeneous depth features from different levels. This strategy usually yields sub-optimal results and misleads the fusion process. Recent works [15, 89, 122, 123] show that dense connection and continual fusion are good choices to learn representations. Inspired by these works, we design DCF block which fully utilizes dense connection to aggregate multi-modal and multi-level representations, as illustrated in Figure 3.3 (a). Commonly, deep fusion networks depend on stacking more layers and increasing channel dimensionality

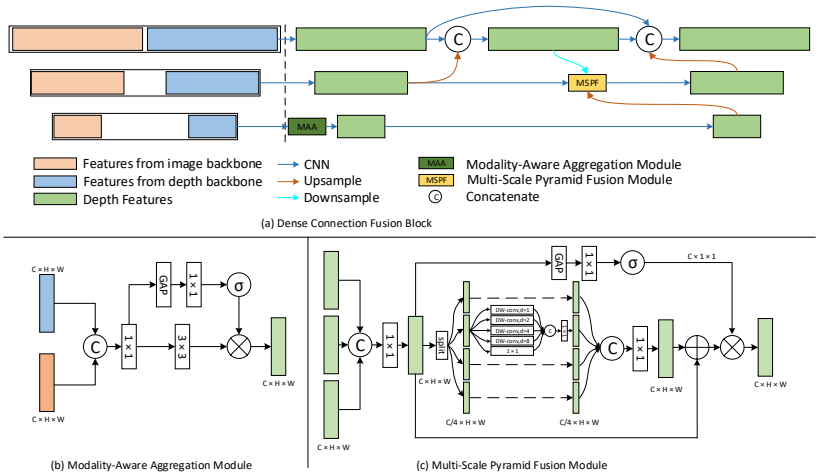
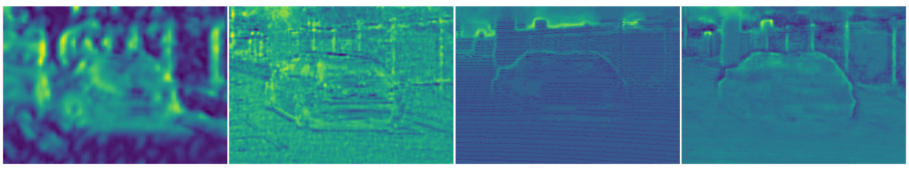


Fig. 3.3: **Illustration of the main modules building CDCNet.** (a) Dense Connection Fusion Block. (b) Modality-Aware Aggregation Module. (c) Multi-Scale Pyramid Fusion Module. Best viewed in color.

to get better results. In contrast, our model adopts a shallow structure and a small number of channels for feature aggregation. Consequently, apart from the standard top-to-down scheme with skip connections, we add an intermediate feature in the aggregation space to compensate for the shallow architecture. The intermediate feature combines features from the same and the higher level at different sizes. Instead of using feature concatenation or sum operation, we also design a modality-aware aggregation module to exploit the discriminative information from the heterogeneous image and depth features, described hereafter.

3.2.2 Modality-Aware Aggregation Module

Different modalities have different attributes to exploit for feature aggregation; therefore, the critical factor for multi-modal fusion consists of exploiting the valuable information from each of the modalities. Image features contain rich semantic information, yet depth features represent strong distance-perceptive information. Most image and depth feature fusion approaches use concatenation operation, which fails to exploit more information from multiple modalities. Hence, we propose a modality-aware aggregation module (Figure 3.3 (b)) which aims at enhancing



(a) high-level (b) middle-level (c) low-level (d) output

Fig. 3.4: **Visualization of multi-level feature maps.** From left to right, (a) high-level, (b) middle-level and (c) low-level feature maps, followed by (d) output of the MSPF module.

multi-modal representation learning. Concretely, given input image features and depth features $F_I, F_D \in \mathbb{R}^{c\times h\times w}$ in each module, we first concatenate them as $F_{cat} \in \mathbb{R}^{2c\times h\times w}$, then use $conv1\times 1$ to smooth F_{cat} and get $F'_{cat} \in \mathbb{R}^{2c\times h\times w}$, Global Average Pooling, $conv1\times 1$ and $sigmoid()$ functions, in order to obtain the modality-aware vector $w \in c\times 1\times 1$ from multi-modal features, which can be formalized as:

$$w = \sigma(conv1\times 1(GAP(conv_{1\times 1}(F_I, F_D)))) \tag{3.1}$$

For F'_{cat}, $conv3\times 3$ are used to get $F_{coarse} \in \mathbb{R}^{c\times h\times w}$. The enhanced features are obtained as:

$$F_M = w \bigotimes conv_{3\times 3}(conv_{1\times 1}(F_I, F_D)) \tag{3.2}$$

Then, we get the modality-aware integrated result $F_M \in \mathbb{R}^{c\times h\times w}$.

3.2.3 Multi-Scale Pyramid Fusion Module

For multi-level features fusion as well, most existing works [9, 12, 15, 18, 19] make use of concatenation or sum operation, which weakens the representation capability of cross-level features. For our task, high-level features have more semantic information while low-level features have more texture information and sparse depth representation, as shown in Figure 3.4(a,b,c). To aggregate multi-level features effectively, we construct a multi-scale pyramid fusion module embedded into the DCF block, as illustrated in Figure 3.3(c). Our motivation arises from Atrous Spatial Pyramid Pooling (ASPP) [124] that uses multiple branches to extract multi-scale features. However, ASPP introduces many parameters and high computational

overhead. Thus, inspired by [125], we construct MSPF in a lightweight way. Suppose $F_H, F_M, F_L \in \mathbb{R}^{c\times h\times w}$ represent the upsampled high-level feature, the middle-level feature, and the downsampled low-level feature, respectively, which are inputs of the MSPF module. We first apply a 1×1 convolution to perform channel pooling for the concatenated multi-level features and get $F_0 \in \mathbb{R}^{c\times h\times w}$. Then, we equally split F_0 into four feature maps $F_1, F_2, F_3, F_4 \in \mathbb{R}^{c/4\times h\times w}$ along the channel dimension, as

$$\begin{aligned} F_0 &= conv_{1\times1}(F_H, F_M, F_L) \\ F_1, F_2, F_3, F_4 &= Split(F_0) \end{aligned} \tag{3.3}$$

Then, for each sub-portion F_i of the original feature map F_0, we apply four 3×3 depth-wise separable convolutions with dilation rates of 1, 2, 4, 8 and 1×1 convolution to implement multi-scale learning and get $F_i^1, F_i^2, F_i^3, F_i^4, F_i^5 \in \mathbb{R}^{c/4\times h\times w}, i = 1, 2, 3, 4$, then we use two consecutive 1×1 convolutions to merge the feature maps $F_i^M \in \mathbb{R}^{c\times h\times w}$ and $F^M \in \mathbb{R}^{c\times h\times w}$:

$$\begin{aligned} F_i^1 &= conv_{1\times1}(F_i) \\ F_i^2 &= conv_{3\times3}^{d=1}(F_i) \\ F_i^3 &= conv_{3\times3}^{d=2}(F_i) \\ F_i^4 &= conv_{3\times3}^{d=4}(F_i) \\ F_i^5 &= conv_{3\times3}^{d=8}(F_i) \\ F_i^M &= conv_{1\times1}(F_i^1, F_i^2, F_i^3, F_i^4, F_i^5) \\ F^M &= conv_{1\times1}(F_1^M, F_2^M, F_3^M, F_4^M) \end{aligned} \tag{3.4}$$

where, $conv_{3\times3}^{d=i}$ denotes 3×3 depth-wise atrous convolution with dilation rate of i.

In the end, we add a residual connection and leverage channel attention [126] to refine the output features, as

$$\begin{aligned} F^M &= F^M + F_0 \\ F^{out} &= F^M \bigotimes \sigma(conv_{1\times1}(GAP(F_0))) \end{aligned} \tag{3.5}$$

where $F^{out} \in \mathbb{R}^{c\times h\times w}$ is the final refined feature.

By splitting features and implementing multi-scale learning with depth-wise

separable convolutions separately, we dramatically cut down computational complexity and reduce the number of parameters. Moreover, the aggregate results embed semantic and texture information from multi-level features, as shown in Figure 3.4(d).

3.2.4 Loss Function

To learn accurate prediction of dense depth maps, we train our network to minimize mean squared error (MSE) and mean absolute error (MAE) losses [12]. A multi-stage and multi-weighted loss function L is the combination of three parts:

$$\begin{aligned} L =& \omega \sum_{i=1}^{N} (L_2(D_i^2, \hat{D}_i^2) + L_1(D_i^2, \hat{D}_i^2)) + \\ & \omega \sum_{i=1}^{N} (L_2(D_i^1, \hat{D}_i^1) + L_1(D_i^1, \hat{D}_i^1)) + \\ & \sum_{i=1}^{N} (L_2(D_i^0, \hat{D}_i^0) + L_1(D_i^0, \hat{D}_i^0)) \end{aligned} \tag{3.6}$$

where N represents the set of valid pixels. D^2, D^1, D^0 denote the predicted depth maps from DCF blocks 0, 1 and 2 respectively, and $\hat{D}^2$, $\hat{D}^1$, $\hat{D}^0$ the corresponding semi-dense ground truth maps. Following [12], we set ω to 1 for the first 6 epochs, then decimating it to 0.1 for 11 epochs, and finally disabling it ($\omega = 0$) until the end of the training procedure.

3.3 Experimental Results

In this section, we evaluate our method and compare it to state-of-the-art solutions on the KITTI depth completion benchmark [1, 7]. we adopted the official evaluation protocol from the KITTI to evaluate our model, i.e., MAE, RMSE, iMAE, and iRMSE.

3.3.1 Implementation Details

We implement CDCNet using Pytorch and train it with a single NVIDIA RTX 3090 GPU. All the parameters are optimized using Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). The learning rate is initialized to 0.001 and multiplied by 0.5 every 5 epochs. A weight decay factor is set to 0.0002. The network is trained for 30 epochs using a batch

size of 6 samples. Training images are cropped to a resolution of 1216×352 pixels. The experiments in the ablation study are carried out by training CDCNet on 10000 samples from the training set and by evaluating on the validation split.

3.3.2 Comparison with state-of-the-art

We compare our model to the state-of-the-art methods published on the KITTI depth completion benchmark. Table 3.1 shows quantitative results retrieved from the online leaderboard. We report the comparison of our method with others in terms of parameters, accuracy and runtime. The number of parameters and runtime are partially taken, respectively, from [127] and [128]. Our lightweight network, CDCNet, outperforms most previous methods under the primary evaluation metric RMSE and achieves results comparable with those by state-of-the-art models. In particular, CDCNet achieves accuracy close to GuideNet [85], CSPN++ [17], NLSPN [19], MDANet [15], with respectively 1.3%, 3.0%, 3.4% and 28% of their total parameters. Compared with MSG-CHN [12] which inspires our method, CDCNet gets better results by using only 69% of its parameters. Due to the diversity of hardware platforms used by each method, performing a fair comparison for what concerns runtime is not trivial. Nevertheless, these results still suggests that our method is faster than most state-of-the-art methods. SPN-based methods [17, 18, 19] slow down their inference time because of the iterative spatial propagation step. PwP [129], DeepLiDAR [13], GuideNet [85] and PENet [130] adopt multi-branch, heavy backbones – i.e., ResNet – which are time-consuming. In contrast, CDCNet takes shorter inference time, despite it processes data in a stacked manner.

For what concerns the main competitor inspiring our work, i.e. MSG-CHN [12], for a fair comparison we use the authors' code and measure its runtime on the same hardware platform used by CDCNet. Our model runs in 0.03 seconds, slower than MSG-CHN [12], because of the feature aggregation modules we introduced. However, this is compensated with higher accuracy and fewer parameters. In summary, CDCNet gets competitive results with clearly fewer parameters on the KITTI depth completion benchmark.

Figure 3.5 reports a qualitative comparison between results yielded by state-of-

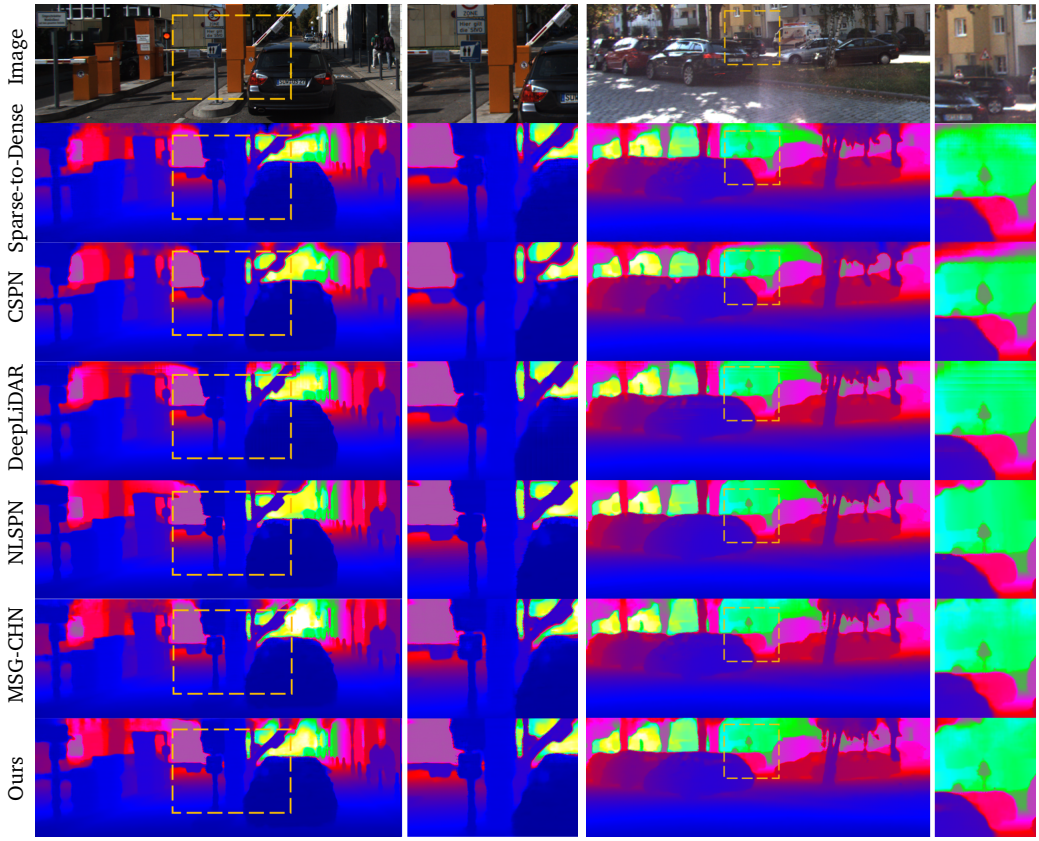


Fig. 3.5: **Qualitative comparison with state-of-the-art methods.** From top to bottom: RGB image, results of Spare-to-Dense[10], CSPN[18], DeepLiDAR[13], NLSPN[19], MSG-CHN[12], and Ours, respectively.We zoom-in the yellow dotted regions at the right.

the-art methods and ours, with the latter being in the last row. Our dense connection fusion strategy, which can efficiently exploit high-level semantic and low-level context information, yields accurate depth maps, preserves finer details on complex structure boundaries and recovers more accurate contours for thin structures in faraway scenes.

3.3.3 Ablation Study

In this section, we demonstrate the effectiveness of the components proposed in this paper.

Impact of Modality-Aware Aggregation Module. By surveying most of the works for KITTI depth completion, concatenation and sum emerge as the dominant image and depth fusion strategies [9, 10, 12, 13, 14, 15, 17, 18, 19, 118, 130]. Besides the methods on the KITTI benchmark, [133] proposed a gated fusion for

Methods	Parameters (M)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)	runtime (s)	Platform
Sparse-to-Dense [10]	-	814.73	249.95	2.80	1.21	0.08	Tesla V100
PwP [129]	29.10	777.05	235.17	2.42	1.13	0.10	Tesla V100
FusionNet [14]	2.50	772.87	215.02	2.19	0.93	0.02	RTX 2080Ti
FuseNet [131]	1.90	752.88	221.19	2.34	1.14	0.09	-
NConv [132]	0.36	829.98	233.26	2.60	1.03	0.02	Tesla V100
DeepLiDAR [13]	53.40	758.38	226.50	2.56	1.15	0.35	RTX 2080Ti
CSPN [18]	-	1019.64	279.46	2.93	1.15	1.00	Titan X
CSPN++ [17]	28.80	743.69	209.28	2.07	0.90	0.20	Tesla P40
NLSPN [19]	25.80	741.68	199.59	1.99	0.84	0.13	RTX 2080Ti
PENet [130]	133.70	730.08	210.55	2.17	0.94	0.16	RTX 2080Ti
GuideNet [85]	63.30	736.24	218.83	2.25	0.99	0.14	GTX 1080Ti
ACMNet [24]	4.90	744.91	206.09	2.08	0.90	0.35	RTX 2080Ti
MDANet [15]	3.07	738.23	214.99	2.12	0.99	0.03	Tesla P100
MSG-CHN [12]	1.25	762.19	220.41	2.30	0.98	0.01	RTX 3090
CDCNet (ours)	0.87	738.26	216.05	2.18	0.99	0.03	RTX 3090

Tab. 3.1: **Quantitative results on the KITTI test set.** We report the amount of parameters, standard evaluation metrics and runtime for state-of-the-art models and CDCNet.

Component	RMSE (mm)	Parameters (K)
Sum	879.81	672
Concat	874.01	727
Gated	882.26	699
MAA	870.39	730

Component	RMSE (mm)	Parameters (K)	Memory (GB)	Speed (ms)
Sum	876.46	616	3.499	18.063
Concat	874.01	727	3.600	18.673
ASPP	870.56	1178	4.286	23.923
MSPF	863.65	695	4.007	27.830

Tab. 3.2: **Ablation study on MAA (left) and MSPF (right) modules.** Sum denotes feature sum operation; Concat denotes feature concatenation operation; Gated denotes Gated Fusion. Parameters, Memory and Speed refers to the entire network processing.

dense image and depth feature fusion. To measure the impact of the MAA module on the final accuracy, we replace it with the alternatives mentioned above in these experiments. It is worth noting that the remaining components are kept unchanged; only the fusion module on which the comparison focuses on changes. From Table 3.2, on the left, we can observe that our fusion strategy yields better results compared to alternative methods, with a limited increase in the number of parameters. Gated fusion results are the worst for this task since this strategy is designed for dense feature fusion, whereas depth features are usually very sparse in completion task.

Impact of Multi-Scale Pyramid Fusion Module. To validate the effectiveness of MSPF, we compare the performance achieved by CDCNet when using it or when it is replaced by sum, concatenation or ASPP alternatives. As shown in Table 3.2, on the right, the results demonstrate that the lightweight MSPF module can achieve the best performance. However, this improvement comes at the expense of speed.

Yet, when directly compared with ASPP, MSPF introduces fewer parameters and requires fewer GPU memory thanks to feature splitting and depth-wise convolutions.

3.4 Conclusion

In this work, we proposed a lightweight yet effective cascade dense connection fusion network, CDCNet. By stacking the dense connection fusion blocks, image and depth features are aggregated effectively in a progressive manner. We employ a modality-aware aggregation method to enhance the fusion of image and depth features. Then, a lightweight multi-scale learning module boosts multi-level feature fusion.

We evaluate CDCNet on the KITTI depth completion dataset achieving competitive results compared to state-of-the-art methods, yet using much fewer parameters.

CHAPTER 4

Lightweight Self-Supervised Depth Estimation with LiDAR Data

Contents

The content of this chapter has been presented at the 34th British Machine Vision Conference (BMVC 2023) - "Lightweight Self-Supervised Depth Estimation with few-beams LiDAR Data" [134].

4.1 Introduction

Despite accurate results achieved by supervised depth prediction methods through the years [19, 20, 23, 48, 54, 135, 136, 137, 138, 139, 140], two main shortcomings still limit the deployment of these approaches as a mature technology. On the one hand, dense ground truth depth annotation is needed to train completion networks. To obtain such data, manual labor [7] is necessary to aggregate several scans performed over time, possibly by a LiDAR sensor sensing the higher density of points possible. Self-supervised depth completion approaches [10, 69, 70, 71] try to soften this constraint by learning from monocular videos and sparse LiDAR how to infer dense depth maps. On the other hand, there is the much higher cost of LiDAR sensors compared to conventional color cameras: indeed, this scales with the density of measurements the sensor can deliver, with 64-beams LiDARs – and more recent, 128-beams devices – costing tens of thousands of dollars. At the same time, cheaper solutions characterized by much fewer emitters (e.g. 4-beams) exist, at the expense of making the completion task even more challenging. On this track, estimating depth from *few-beams* LiDAR data [73, 141], possibly in a self-supervised manner, represents the cheapest chance to develop a framework capable of densifying sparse depth measurements and requiring low-cost depth sensors to deploy it. Nonetheless, solutions proposed so far [73, 141] still rely on very complex CNNs, counting tens of millions of parameters and thus putting some constraints on the hardware capabilities required for deployment.

Common strategies to collect depth data rely on active sensors, precisely measuring the distance of objects in the scene by perturbing them through some signals. However, they only provide sparse depth information, resulting in many empty regions for which no measurement is available. For instance, the Velodyne HDL-64e LiDAR used in the KITTI dataset provides accurate, yet sparse depth data with a density lower than 6% compared to the image resolution. This fact makes it hard

to tackle downstream perception tasks such as detection, semantic segmentation, or instance segmentation. Fortunately, many methods have been proposed and remarkable progress has been achieved in depth prediction tasks with the help of deep learning.

Supervised depth prediction methods, such as depth completion [12, 13, 17, 25, 84] and depth estimation [33, 36, 52, 120] requires high-quality and large-scale dense depth labels, which are expensive and hard to get. Self-supervised monocular depth estimation [10, 56, 57, 58, 62, 63] with multi-view images are trained to predict the dense depth and camera ego-motion without label and utilize the re-projection photo-metric loss as supervision. Nevertheless, both methods have some shortcomings. Supervised methods call for dense annotated depth maps and are hard to cope with the non-annotated scenes. Self-supervised methods suffer from some significant challenges, such as undesirable prediction on low-texture regions. Reprojection loss recovers the depth to an ambiguous scale. Thus it would be attractive to combine the strength of self-supervised depth estimation and the observed sparse depth measurements. [71, 72, 73, 142] extend self-supervised training scheme with sparse depth measurements. These approaches combine the advantages of sparse measurements and self-supervised monocular depth estimation which can ground the predictions to metric scale and train the model without dense depth labels. while [71, 72] use sparse convolution [7]. These methods failed to deal with monocular video with sparse measurements in a lightweight and effective way. And all methods stress the consistency between the prediction and input in the sparse depth domain, they ignore the outlier in the inputs.

Therefore, we take a further step toward inexpensive solutions for densifying sparse LiDAR data developing a lightweight yet effective self-supervised network for this task. Our proposal involves the use of a multi-stage architecture that is designed to effectively utilize the guidance provided by color images during the densification process. For this purpose, we revise Sparsity-invariant CNNs [7] and introduce a novel layer called Guided Sparsity-invariant CNNs, capable of effectively processing the contextual information provided by dense guidance. Moreover, to further improve the accuracy of our model, we implement a Distance-Dependent

Outlier Mask, capable of mitigating the impact of outliers in the sparse data on the resultant dense depth map.

In this work, we propose a lightweight yet effective self-supervised network processing few-beams LiDAR data and a single image. It counts as few as ∼600K parameters. At the core of our architecture, we propose a Guided Sparsity-invariant CNN block, which can deal with sparse data to produce depth features under the guidance of dense color images or depth maps. To cope with outliers in the sparse input data, we introduce a Distance-Dependent Outlier Mask to mitigate the impact on the final predictions. We evaluate our framework processing data from cheap (4-beams) and expensive (64-beams) LiDAR sensors, achieving state-of-the-art performance in the former case and yielding results equivalent to existing models in the latter case, despite utilizing only about 2% of the parameters required by those models.

4.2 Self-Supervised Depth Estimation from LiDAR data

This section describes our proposal for a self-supervised depth estimation network from color images and sparse LiDAR measurements. The proposed self-supervised framework aims at predicting a dense depth map $\hat{D} \in \mathbb{R}^{H \times W \times 3}$ from monocular image $I \in \mathbb{R}^{H \times W \times 3}$ and the sparse 4-beams LiDAR depth map $S \in \mathbb{R}^{H \times W}$, which is aligned with I. We formulate this task as a self-supervised learning problem, obtaining supervision from color images in a video and the very same input depth points. Figure 4.1 provides an overview of the architecture of our framework.

4.2.1 Self-Supervised Depth Estimation from few-beams LiDAR

We introduce our framework for self-supervised completion, which consists of two main networks to predict depth and ego-motion [56].

Lightweight Multi-stage DepthNet: The DepthNet takes a single image I, and the corresponding sparse depth map S as inputs to progressively recover a dense depth map $\hat{D}$. It follows a multi-stage design, common in literature [11, 12, 13, 15, 85, 130], consisting of one image backbone and three cascade depth estimation networks. The former extracts multi-scale color features through convolutions and

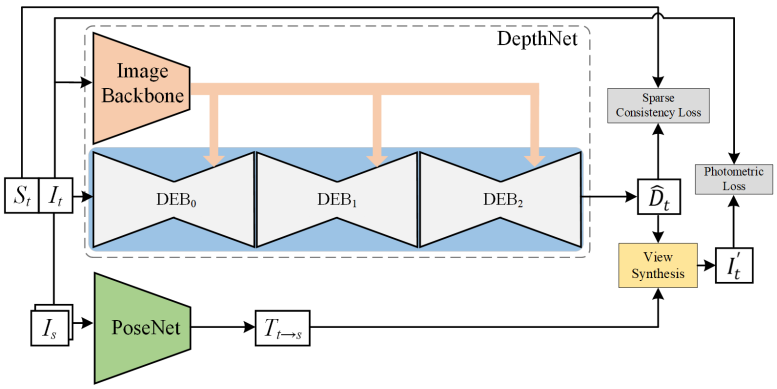


Fig. 4.1: **Overview of our framework:** A DepthNet processes a single image and corresponding LiDAR data to predict a dense depth map. A PoseNet estimates the camera ego-motion from two images during training.

downsampling operators, encoding semantics and texture as guidance to recover dense depth. These features are then fed to the three cascade Depth Estimation Blocks (DEB), namely DEB_0, DEB_1, DEB_2, from left to right in Fig. 4.1, respectively. The three blocks are compact encoder-decoder networks, sharing the same architecture for the decoder, processing sparse depth points at a quarter (S_0), half (S_1), and full resolution (S_2) respectively – with S_0, S_1 being downsampled from S_2, i.e., S points on the image plane – as well as color features. Each encoder relies on Guided Sparse Convolutions – introduced in the remainder – having 32 output channels each. The decoders predict outputs at the exact resolution as the original input to the specific DEB block, processing image features from the image backbone. Residual connections [11, 143] integrate the results by the three after upsampling to full resolution.

PoseNet: Inferring camera ego-motion is essential for learning depth estimation from videos in a self-supervised manner. Thus, following [57], our PoseNet uses an ImageNet pre-trained ResNet18, taking two stacked color images as input to infer their 6-DoF relative pose. This network is needed at training time only.

4.2.2 Guided Sparsity-Invariant Convolution

Although most of the existing approaches [11, 12, 13, 15, 85, 130] rely on standard CNNs to extract sparse depth features through dedicated branches, Uhrig *et al.* [7] demonstrated that this approach is sub-optimal when dealing with highly-sparse data, and proposed Sparsity-invariant CNNs (SCNNs) to handle it better. However, SCNNs

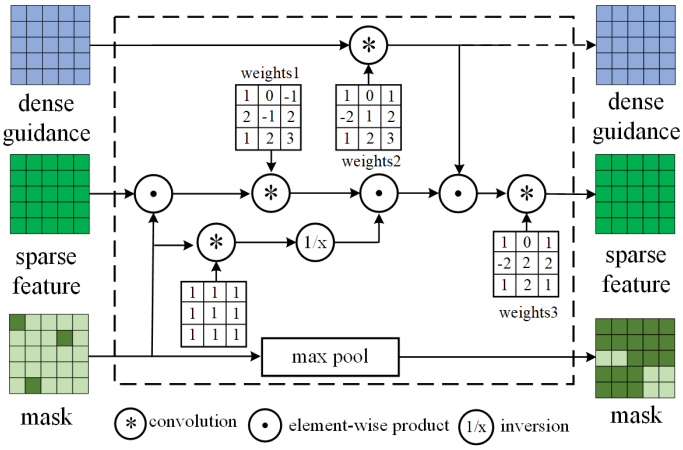


Fig. 4.2: **Guided Sparsity-invariant CNN (GSCNN).**

do not exploit any guide from color images usually coupled with the sparse data we aim at densifying, a powerful cue seldom ignored when available [11, 12, 13, 15, 85, 130]. To overcome this lack of the original SCNNs, we revise it to exploit additional dense guidance, as shown in Fig. 4.2.

Specifically, we propose Guided Sparsity-Invariant CNNs (GSCNNs) to overcome one main limitation of SCNNs, which struggle to recover sharp object boundaries due to the lack of awareness of semantic and dense structural cues that are available on RGB images instead. For this purpose, we introduce dense guidance d as an additional input to SCNNs, which will guide the propagation process of sparse data s within the network. The dense guidance can be color image, depth maps, or multi-channel depth features, and the input sparse feature can be LiDAR data or multi-channel depth features. A standard convolution operator processes these features, then multiplied to sparse features processed according to the standard SCNN design – i.e., a binary validity mask m is used to identify the meaningful features of the sparse data from those extracted out of invalid inputs – and then a final convolution produces the enhanced, sparse features output of the GSCNN layer.

This revised design keeps the merits of SCNNs to deal with sparse data s more effectively than CNNs while complementing its lack of semantic knowledge with the dense guide d. GSCNNs can be formalized as follows:

$$f_i(d, s, m) = \sum_{j \in \Omega(i)} w_j^3 [(\sum_{j \in \Omega(i)} w_j^2 d_j)(\frac{\sum_{j \in \Omega(i)} m_j \, s_j \, w_j^1}{\sum_{j \in \Omega(i)} m_j + \epsilon} + b)] \tag{4.1}$$

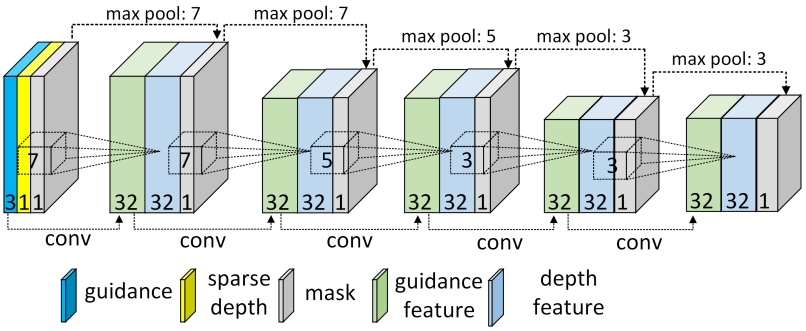


Fig. 4.3: **DEB encoder.** Five GSCNN layers extract sparse depth features each with decreasing kernel sizes from 7×7 to 3×3.

with $\Omega(i)$ being the convolution window centered in i, w^1, b weights and bias already present in standard SCNNs and w^2, w^3 the additional weights used to process d.

We use GSCNNs to extract sparse depth features in DEB encoders. For DEB_0, we stack five GSCNN layers, as shown in Fig. 4.3, using the color image as guidance alone. In DEB_1 and DEB_2, we use GSCNN layer in the first layer only, this time guided by dense output predicted by the previous DEB block, i.e. DEB_0 and DEB_1 for the two, respectively.

4.2.3 Loss function

Following the literature [71, 73, 142, 144], our model is trained using three loss terms that are optimized jointly:

$$L_{total} = \alpha L_{ph} + \beta L_{sm} + \gamma L_{sd} \tag{4.2}$$

with L_{ph}, L_{sm}, L_{sd} denoting the photometric consistency, smoothness, and sparse depth consistency losses, weighted by α, β and γ, respectively. Following [57], we compute these terms on intermediate depth predictions, i.e., on the output of each DEB block upsampled to the original input resolution.

Photometric Consistency Loss. Given the camera intrinsic matrix K, we synthesize the target image I'_t by warping the source image I_s according to the estimated depth and relative poses. As in [55, 57], we evaluate the pixel-level similarity between I'_t and the real target image I_t using a combination of an L1 pixel-wise loss

term and the Structural Similarity (SSIM) [145] term:

$$L_{ph}(I_t, I'_t) = \alpha \frac{1 - \text{SSIM}(I_t, I'_t)}{2} + (1 - \alpha) \parallel I_t - I'_t \parallel, \tag{4.3}$$

We adopt auto-masking [57] to filter out static pixels and the occluded region.

Smoothness Loss. We enforce a smoothness constraint on the dense depth maps by utilizing texture information from the input color image [57]:

$$L_{sm} = \mid \partial_x d^* \mid e^{-|\partial_x I_t|} + \mid \partial_y d^* \mid e^{-|\partial_y I_t|}, \tag{4.4}$$

with ∂_x, ∂_y being gradients along x and y direction, and $d^* = \hat{d}_t / \overline{\hat{d}_t}$ normalized inverse depth.

Sparse Depth Consistency Loss. We enforce consistency between densified and sparse depth using the scale-invariant [33] depth loss:

$$L_{si} = \frac{1}{2n^2} \sum_{i,j} \Big((\log y_i - \log y_j) - (\log y_i^* - \log y_j^*) \Big)^2 \tag{4.5}$$

with y and y^* being the predicted and input depth over the whole depth map space Ω, respectively, and n the number of pixels.

The raw sparse depth data contains outliers primarily due to the displacement between the LiDAR and the color camera. This misalignment causes the projection of some background points to overlap with foreground objects, as shown in Fig. 4.4. This fact would yield background points to emerge on the foreground objects in the predicted dense maps, causing inaccuracy near the depth discontinuities.

To avoid this behavior, we design a Distance-Dependent Outlier Mask M by setting a threshold σ on the discrepancy D_δ between prediction $\hat{D}$ and sparse depth measurements S. Such a threshold is dynamic, it varies in the different distance ranges over the sparse depth domain Ω, since a relatively more significant error is tolerable when predicting a farther depth value [146, 147].

To ease convergence, we empirically first set $\sigma = 4.0$ for the first 2 epochs:

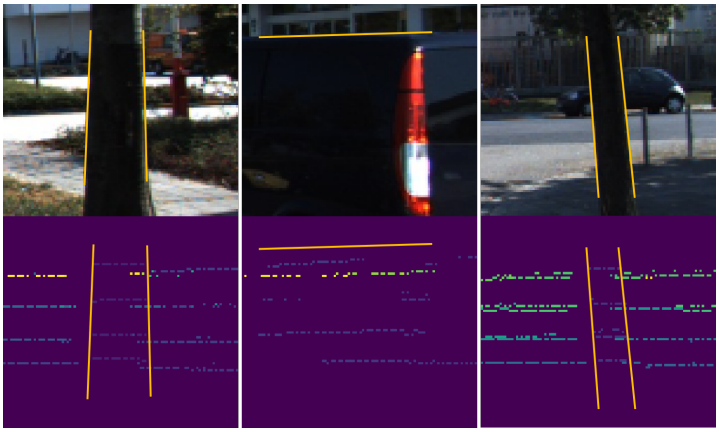


Fig. 4.4: **Outliers on depth data.** Three images and corresponding LiDAR points, with overlapping background and foreground points.

$$M(x) = \begin{cases} 1 & \text{if } D_\delta(x) < \sigma \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

Then, we set multiple thresholds σ_i according to different depth ranges:

$$M(x) = \begin{cases} 1 & \text{if } D_\delta(x) < \sigma_1, \forall\ \hat{D}(x) < 5 \\ 1 & \text{if } D_\delta(x) < \sigma_2, \forall\ \ 5 \leq \hat{D}(x) < 10 \\ 1 & \text{if } D_\delta(x) < \sigma_3, \forall\ 10 \leq \hat{D}(x) < 20 \\ 1 & \text{if } D_\delta(x) < \sigma_4, \forall\ 20 \leq \hat{D}(x) < 30 \\ 1 & \text{if } D_\delta(x) < \sigma_5, \forall\ 30 \leq \hat{D}(x) \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

with $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$ set to 0.2, 0.4, 0.8, 1.0, 2.0. We will analyze the effect of a fixed threshold σ over the whole depth range in the ablation study.

The overall, sparse depth consistency loss L_{sd} is then defined as:

$$L_{sd} = \omega \sum_{x \in \Omega} L_{si}(M(\hat{D}_0(x), S(x))) + \omega \sum_{x \in \Omega} L_{si}(M(\hat{D}_1(x), S(x))) + \sum_{x \in \Omega} L_{si}(M(\hat{D}_2(x), S(x))) \tag{4.8}$$

with $\hat{D}_0(x), \hat{D}_1(x), \hat{D}_2(x)$ being the predicted depth maps, S the sparse input

depth, M an outlier mask used to ignore them – described in the remainder – and ω a hyper-parameter to control the impact of the loss on intermediate predictions. Specifically, we use a multi-stage training scheme by setting $\omega = 1$ for 10 epochs and then reducing it to 0.5 until convergence.

4.3 Experimental Results

We introduce our experiments on two tasks: 1) self-supervised depth estimation from color images and few-beam LiDAR data and 2) depth completion without groundtruth.

4.3.1 Datasets.

We focus on self-supervised depth estimation with LiDAR data in the outdoor environment. KITTI dataset [1] is popularly used in depth estimation. For what concerns the few-beams LiDAR setting, we follow [73] and evaluate our method on the Eigen split [34] of the KITTI original dataset[1] by uniformly sampling the sparse 4-beams data from original 64-beams LiDAR data [73, 148]. Regarding the standard depth completion setting – i.e., with 64-beams LiDAR – we test on the KITTI Depth Completion validation set [7].

4.3.2 Implementation Details.

We use PyTorch [149] and train our model with a single NVIDIA RTX 3090 GPU, implemented starting from [73] code base. The sparse depth is normalized in the range $[0, 1]$ before being processed by our model, which predicts multi-scale dense disparity maps, and then brings them back to the metric scale. All the parameters are optimized using Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). The learning rate is initialized to 0.004 and multiplied by 0.5 every 8 epochs. We set the weight decay factor to 0.0002, and the network is trained for 40 epochs using a batch size of 20 samples with input images downsampled to 640×192.

	Method	Input	Train	Parameters	The lower the better				The higher the better		
					Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
(1)	Dorn [146]	M	M+Sup	99M	0.099	0.593	3.714	0.161	0.897	0.966	0.986
	BTS [50]	M	M+Sup	52M	0.091	0.555	4.033	0.174	0.904	0.967	0.984
(2)	MonoDepth2 [57]	M	S	14M	0.109	0.873	4.960	0.209	0.864	0.948	0.975
	MonoDepth2 [57]	M	M+S	14M	0.107	0.849	4.764	0.201	0.874	0.953	0.977
(3)	LEGO [153]	M	M	-	0.162	1.352	6.276	0.252	0.783	0.921	0.969
	PackNet-SfM [58]	M	M	>50M	0.111	0.785	4.601	0.189	0.878	0.960	0.982
	MonoDepth2 [57]	M	M	14M	0.115	0.903	4.863	0.193	0.877	0.959	0.981
(4)	Guizilini *et al.* [141]	M+L	M+L	>50M	0.082	0.424	3.73	**0.131**	0.917		
	FusionDepth[73]	M+L	M+L	26M	0.078	0.515	3.67	0.154	0.935	0.973	0.986
	FusionDepth*[73]	M+L	M+L	26M	0.076	0.490	3.63	0.149	0.934	0.974	0.986
	FusionDepth (Refined Depth)[73]	M+L	M+L	>26M	0.074	**0.423**	3.61	0.150	0.936	0.973	0.986
	Ours	M+L	M+L	**628.53K**	**0.069**	0.476	**3.31**	0.144	**0.943**	**0.975**	**0.987**

Tab. 4.1: **Depth prediction on KITTI Eigen Split.** All methods process 640×192 images. M, S, and L respectively indicate Monocular, Stereo, and Sparse LiDAR data, with Sup referring to supervised training with accurate ground truth. Results for existing methods are directly taken from [73]. * means retrained by ourselves (with better results).

4.3.3 Depth Estimation from few-beams LiDAR

Following [73], we compare our model with methods representative of four main categories: (1) supervised monocular networks [50, 146]; self-supervised monocular networks trained on (2) stereo pairs [57, 150, 151] or (3) monocular videos [56, 57, 152]; (4) self-supervised monocular methods from few-beams LiDAR[73, 141]. Table 4.1 collects the outcome of our experiments. In the case of FusionDepth, the authors employed GDC (post-processing) results as a form of supervision, leading to further improvements in performance. However, it is important to note that GDC requires additional parameters and computational resources. To ensure a fairer comparison, we have deliberately chosen to adopt the FusionDepth results without GDC and prioritize the evaluation of pure self-supervised models. Not surprisingly, methods processing even the few depth points from 4-beams LiDARs notably outperform the others. Among them, our model achieves the best results on the Eigen split, with extremely few parameters.

Moreover, we evaluate the accuracy achieved by our model on another *very-sparse* setting, i.e. by randomly sampling only a few hundred depth points from the sparse

Methods	Params	Samples	Abs Rel	RMSE
Sparse-to-dense [10]	26.1M	100	0.074	4.11
FusionDepth [73]	26M	100	0.074	**4.11**
Ours	**628.53K**	100	**0.072**	4.13
Liao *et al.* [154]	-	225	0.113	4.50
Sparse-to-dense [10]	26.1M	200	0.069	3.92
FusionDepth [73]	26M	200	0.069	**3.92**
Ours	**628.53K**	200	**0.066**	4.01

Tab. 4.2: **Self-supervised depth estimation.** Experiments with randomly sampled LiDAR.

Methods	Params	RMSE	iRMSE	iMAE
Sparse-to-dense [10]	26M	1342.33	4.28	1.64
DPP [69]	$\approx$ 18.8M	1310.03	-	-
VOICED [70]	$\approx$ 6.4M	1230.85	3.84	1.29
SelfDeco [71]	-	1212.89	3.54	1.29
FusionDepth [73]	26M	**1193.92**	3.39	**1.28**
Ours	**628.53K**	1234.75	**3.25**	1.29

Tab. 4.3: **Self-supervised depth completion.** Experiments with 64-beams LiDAR.

LiDAR [10, 73, 154]. Table 4.2 collects the outcome of this experiment. Even in this very challenging scenario, our model yields results close to existing methods while being much more compact.

4.3.4 Depth Completion

Finally, we also evaluate our method when processing denser depth maps provided by a more expensive HDL-64 LiDAR sensor – the one used by the standard KITTI depth completion dataset, yet training in a self-supervised manner. The performance of our model and existing self-supervised solutions on the KITTI completion validation set are reported in Table. 4.3. Despite the much fewer parameters, our lightweight network achieves results comparable with those yielded by state-of-the-art models.

4.3.5 Ablation Study

We conclude with ablation studies to assess the effectiveness of the proposed modules, DEBs, GSCNNs, and the outlier mask. All experiments are conducted with the few-beams LiDAR setting on the Eigen split.

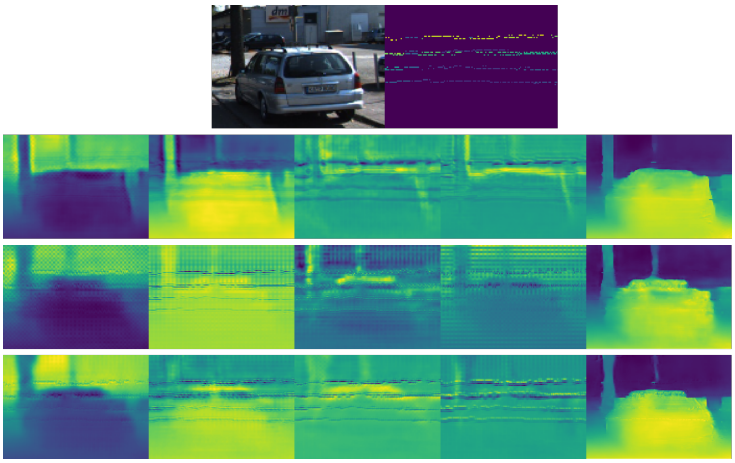


Fig. 4.5: **Ablation studies – qualitative comparison between outputs by using, from top to bottom, by GSCNNs, SCNNs, and CNNs.** The top two inputs consist of image and LiDAR data. The first four columns represent the extracted feature maps, while the remaining columns display the predicted results.

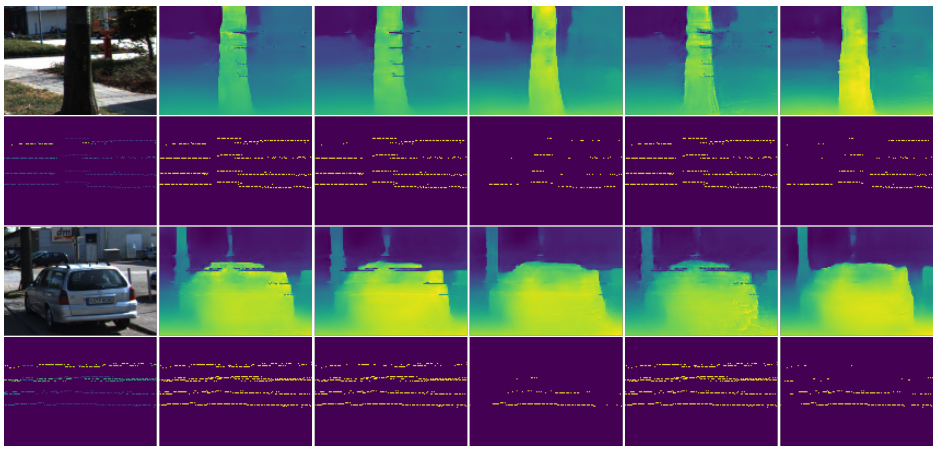


Fig. 4.6: **Ablation studies – qualitative results between different masks.** Depth maps predicted, from left to right, without using any outlier mask, a fixed $\sigma = 4.0$ or 0.2, confidence predicted over the input depth or our strategy.

Cascade Depth Estimation Blocks. Our network progressively recovers dense depth map block by block, with each DEB predicting a dense depth map. To better study the effectiveness of this module, we conducted a study on the impact of the number of blocks regarding performance, speed, and computation cost, as presented

in Table 4.4. All the results indicate that the models run on a single RTX 3090, with an input resolution of 1216×352. Through experimentation with the number of DEBs ranging from 2 to 4, we observed that increasing the number of blocks leads to higher computation costs without consistent performance improvement beyond three blocks. Consequently, an architecture comprising three cascade DEBs is the best suited for our purposes.

DEB Blocks	GFLOPs	Params	FPS	latency	Abs Rel
2	44.3	335K	160.16	0.006s	0.072
3	46.8	468K	123.79	0.008s	0.069
4	47.5	475K	100.97	0.010s	0.075

Tab. 4.4: **Ablation studies.** Comparison between models deploying different numbers of DEBs.

Guided Sparsity-Invariant Convolution. To validate the effectiveness of GSCNN, we compared the performance of three different variants of our framework, obtained by using the proposed GSCNNs, the original SCNNs, or the standard CNNs to build the DEB encoders. From Table 4.5, we can observe that using GSCNNs yields better results on two out of three metrics compared to alternative methods, in particular in terms of RMSE, with only a limited increase in the number of parameters.

To further validate this finding, we visualize the feature maps extracted from images and sparse data in the last block and the predicted results by the three methods in Fig. 4.5. Using GSCNNs (2nd row) allows for extracting much more detailed features, already allowing for distinguishing foreground objects from the background and prediction with clearer boundaries and details. In contrast, SCNNs and CNNs (3rd and 4th rows) extract features exposing grid artifacts and scarce semantic information. Based on the above analysis, GSCNN demonstrates superiority over SCNN and CNN in both quantitative and qualitative comparisons, despite having limited additional parameters.

Distance-Dependent Outlier Mask. To validate the effectiveness of the proposed outlier mask, we compare its performance with alternative approaches [155, 156] – by adding a binary confidence layer in our model to identify the outlier in the sparse input – as well as to the use of a fixed threshold σ for any depth range. Results are

Layers	Params	Abs Rel	Sq Rel	RMSE
CNN	363.971K	0.072	0.463	3.450
SCNN	446.611K	0.070	**0.450**	3.466
GSCNN	628.531K	**0.069**	0.476	**3.312**

Tab. 4.5: **Ablation studies.** Comparison between CNNs, SCNNs[7] and GSCNNs.

reported in Table 4.6. From it, we can notice that our strategy is the only one yielding consistent improvements on any metric. Figure 4.6 shows a qualitative comparison between the dense depth maps predicted according to the different strategies. The absence of any outlier mask (2nd column) produces holes in the foreground objects, like using a fixed threshold $\sigma = 4.0$ (3rd column). A stricter threshold equal to 0.2 (4th column) can alleviate this behavior, yet without significant improvements on the final accuracy according to Table 4.6, while using confidence still cannot prevent holes from appearing in the densified maps (5th column). Our strategy (rightmost column) can remove holes and improve results quantitatively.

Methods	Abs Rel	Sq Rel	RMSE
w/o mask	0.071	0.504	3.492
$\sigma = 4.0$	0.070	0.489	3.429
$\sigma = 0.2$	0.070	0.538	3.434
confidence	0.071	**0.465**	3.472
ours	**0.069**	0.476	**3.312**

Tab. 4.6: **Ablation studies.** Comparison with different masking techniques.

4.4 Conclusion

In this chapter, we have proposed a lightweight architecture for self-supervised depth estimation from sparse depth points and color images. Thanks to the revised Guided Sparsity-Invariant CNNs design, our model can accomplish accurate predictions without the need for over-parametrized layers. Moreover, the proposed Distance-Dependent Outlier Mask prevents outliers in the sparse data from irremediably damaging the predicted dense depth map. Experimental results with multiple settings, i.e. 4-beam, 64-beam, and a few hundred depth points, LiDAR data show that our

model yields state-of-the-art accuracy with a minimal fraction of the parameters used by existing frameworks.

CHAPTER 5

Contrastive Learning for Depth Prediction

Contents

The content of this chapter has been presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop on End-to-End Autonomous Driving: Perception, Prediction, Planning and Simulation (E2EAD 2023) - "Contrastive Learning for Depth Prediction" [157].

5.1 Introduction

Regardless of the adopted setup, little effort in the literature focuses on analyzing the distribution of depth data in a statistical manner. In the real world, the depth changes smoothly within adjacent pixels belonging to the same object's surface. However, when depth prediction models infer the depth, they may output different results, as shown in Fig. 5.1. These models formulate depth prediction as a regression task, and the output value within an extremely small region changes slowly. When a weak model predicts the depth, it is common to find that the result is represented by peaks in a narrow range of depth values, which can be observed from depth predictions and depth histograms in Fig. 5.3.

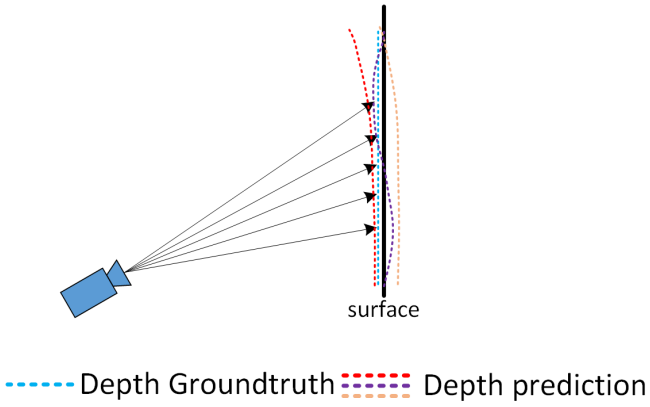
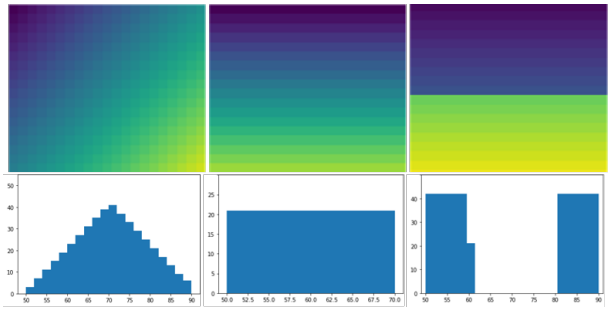


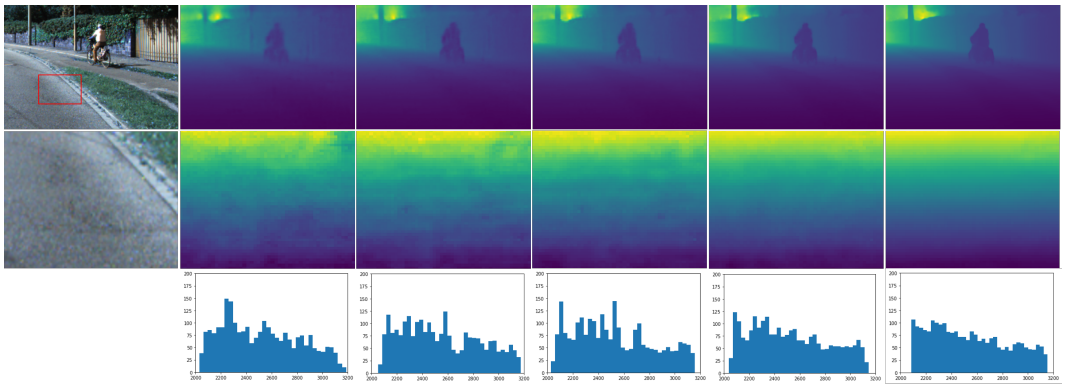
Fig. 5.1: **Imaging system and depth prediction process.** The depth changes smoothly within adjacent pixels belonging to the same portion of the object, while this is not always the case for depth predicted by neural networks. Best viewed in color.

A depth histogram is a graphical representation of the value distribution in a depth map. By focusing on the depth distribution analysis in Fig. 5.3, we can notice how histograms change considerably: indeed, in poor-quality depth maps, depth histogram is usually concentrated in limited intensity values, whereas high-quality



(a) **(b)** **(c)**

Fig. 5.2: **Three synthetic depth maps and corresponding histograms.** (a), (b) are two smooth surfaces and (c) is a depth discontinuity.



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Fig. 5.3: **Illustration of different quality depth maps.** From left to right, rows 1 and 2: (a) the color image (first row) and one region in the next two rows corresponding to the red box in the original image. Columns (b, c, d, e, f) show depth maps of different quality, from worse to better. Row 3 reports depth histograms for the depth maps corresponding to the area within the red box. We took (b-e) from four training stages of CDCNet[11] on the KITTI depth completion task [1].

depth maps have a more regularized distribution spread into a broader interval.

To further confirm this observation, we synthesize three ideal depth maps for two smooth surfaces and one discontinuity – i.e. common structures in the real world – and construct their depth histogram, as shown in Fig. 5.2. We can observe that the distributions of depth values in the histograms are much more regular. Therefore, we argue that regularizing such distributions can yield higher-quality depth maps. Accordingly, by focusing on the structure of objects from a microscopic perspective, any object can be seen as the composition of several small, smooth surfaces interleaved by depth discontinuities, whose depth distributions can be regularized.

Thus, we inquire whether a learning method can regularize the distribution of the depth predictions by a deep network. The recent contrastive learning approaches [94, 96, 158] proposed to contrast positive pairs – i.e., elements sharing the same label – against negative pairs – elements with different labels – in the representations space, which looks suitable for our purpose. Therefore, we introduce a contrastive learning framework tailored for depth prediction. Specifically, we propose a Window-based Contrastive Learning module (WCL) which segments the depth feature maps into non-overlapping windows and computes a contrastive loss only within each one, similarly to how recent Vision Transformers [112] compute self-attention on local windows. It allows us, in a more tractable manner rather than acting on the whole depth features, to contrast the depth values in small regions and expand their distribution locally by constructing positive and negative pairs in the depth representation and enlarging the gap between them. To the best of our knowledge, we are the first to apply contrastive learning for depth prediction, focusing on expanding depth distribution.

In this work, we propose a novel method combining contrastive learning with depth prediction, contrasting depth distribution to improve accuracy. We propose a Window-based Contrastive Learning (WCL) module for depth prediction by learning similarity, forming positive and negative pairs of features, and computing contrastive loss within a window. Experimental results and a detailed ablation study with different depth prediction models demonstrate the effectiveness of our proposal.

5.2 Contrastive learning for Depth Prediction

In this section, we first present the motivation for our work and how WCL can improve depth prediction.

5.2.1 Motivation

In the imaging system, the image is a perspective projection of a 3D scene, and the corresponding depth map reflects the distance between each point in the scene and the viewpoint. Almost all surfaces of observed objects have discrete depth values and adjacent pixels, even within small regions, have similar but different depth values. As shown in Fig. 5.3, the histograms of different depth images

with different accuracy change dramatically, and high-quality depth maps have a more regular depth distribution than low-quality ones. Hence, an intuitive idea to model this assumption consists in regularizing such distribution. For this purpose, applying contrastive learning by clustering the different pixels in the representation space seems a promising strategy for possibly regularizing the depth distribution. Moreover, it has been recently applied in an unsupervised manner [94, 96, 97, 98, 99], thus not making use of labels. Some recent works [99, 101, 159, 160] focus on dense predictions, such as semantic segmentation, and use similarity or affinity between pixels, images, or features to construct contrastive loss for the pixels with the same label that share similar low-level features (color, texture) or high-level representation. Specifically, [159, 160] propose to use the semantic similarities among labeled pixels to contrast representation. Ke *et al.* [101] explore different types of contrastive relationships, such as low-level image similarity and feature affinity in weakly supervised segmentation. Chaitanya *et al.* [99] use contrastive learning at the level of local and global features. All these methods assume that pixels belonging to the same object share the same representation and label since the target task is semantic segmentation. However, this assumption does not hold when facing depth prediction tasks, since even pixels from the same object category may have different representations and depth labels. Indeed, distinguishing the positive and negative pairs in the depth map is challenging, and some of the main issues about deploying a contrastive loss in depth prediction tasks are:

- Discriminating between positive and negative examples for all pixels is challenging. For example, in Fig. 5.4, we can easily define the set (A, B) as a positive pair since the two are close in distance and representation and the set (A, D) as a negative pair being far in distance and representation. However, for the set (B, C), it is hard to define whether they are positive or negative pairs since they are close in distance but belong to different objects.

- Constructing and computing a global relationship graph for all pixels is highly resource-consuming. For instance, if we consider an H $\times$ W image, its relationship map results in size HW $\times$ HW, thus forming positive and negative pairs and contrasting them yields massive memory footprints, computation,

and time.

- It is hard to contrast further the long-range sets, such as set (A, D), even in low-quality maps since there is already a significant distance between the two points.

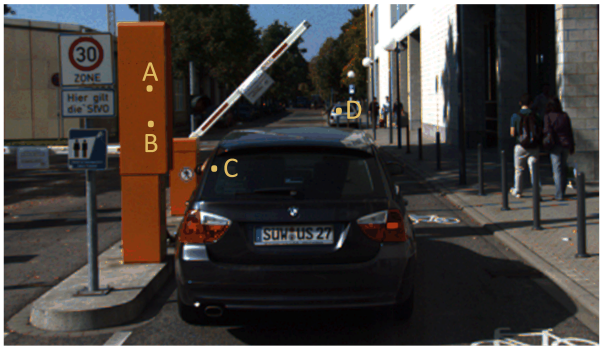


Fig. 5.4: **Example of different points in the scene.** A and B are two points on the same, front-parallel surface, while C and D are points on two distinct cars. Although both A-B and C-D pairs of points belong to similar objects, A-B exposes stronger similarity for the depth prediction task – i.e., they are very close, while C-D points are at very different distances.

For the reasons outlined, employing contrastive learning on the whole image is challenging. Purposely, we introduce the concept of local similarity in our method. As already pointed out, the output depth of a deep network changes smoothly within a small region. Therefore, a pixel has more similarities with its surrounding pixels. The closer the two pixels are, the stronger the similarity. For instance, The similarity between A and B is stronger than the similarity between A and others in Fig. 5.4. To deal with these issues, inspired by some works [42, 112, 113], we propose for depth prediction a cost-effective window-based approach for contrastive learning. We limit the similarity calculation and construct the positive and negative pairs within small regions rather than on the entire feature map, which is a more reasonable and resource-saving strategy. Thus, we set pairs with strong similarity as positive pairs and ones with weak similarity as negative pairs. Enlarging the gap between them also makes depth distribution better regularized and yields more accurate results.

5.2.2 Window-based Contrastive Learning (WCL)

Fig. 5.5 depicts the architecture of the proposed WCL module. It segments one H×W×C feature map into multiple tiles of the same size, each containing N × N elements. In our approach, these windows are the domain where contrastive learning comes into play.

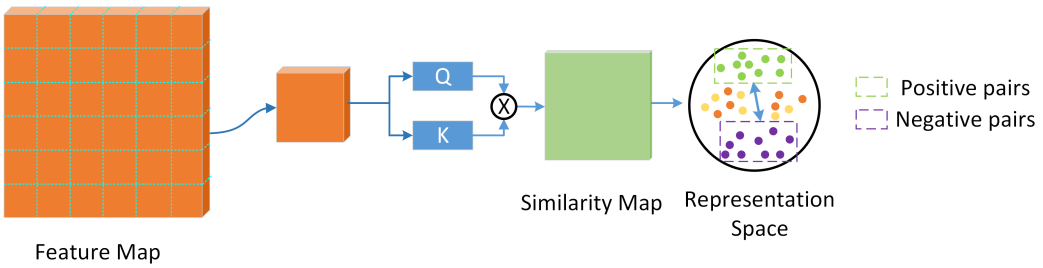


Fig. 5.5: **Illustration of WCL module.** Our module segments one $H \times W \times C$ feature map into multiple tiles of the same size, each containing $N \times N$ elements. Then, it computes a similarity map and constructs positive and negative pairs within the window. By enlarging the gap between positive and negative pairs, the features representation becomes more meaningful of the depth distribution in the scene.

Given a generic features map $F \in \mathbb{R}^{H \times W \times C}$, for instance, the output of a convolutional layer, we partition it into windows $X \in \mathbb{R}^{N \times N \times C}$. For each, we extract query $Q \in \mathbb{R}^{N \times N \times \frac{C}{2}}$ and key $K \in \mathbb{R}^{N \times N \times \frac{C}{2}}$ features by linear projections of the input X as

$$\begin{aligned} Q &= XW_Q \\ K &= XW_K \end{aligned} \tag{5.1}$$

The similarity map $T \in \mathbb{R}^{N^2 \times N^2}$ between each element in the window is computed by means of dot product and normalization as

$$T = \frac{Q^T}{\|Q\|} \frac{K}{\|K\|} \tag{5.2}$$

We employ the exponential function $exp()$ to make the similarity map non-negative. Then, we sort T in descending order:

$$T' = sort(T) \tag{5.3}$$

and use contrastive learning to enlarge the gap between positive and negative pairs

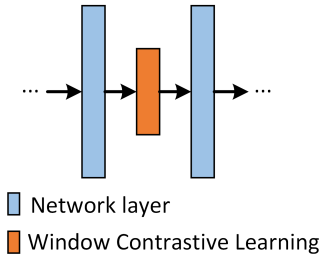


Fig. 5.6: **WCL positioning.** Illustration of a network with a WCL module embedded in between two conventional layers.

in the representation space. We sample the first N_1 of $T^{'}$ as positive pairs P_T, and the last N_1 negative pairs as N_T to compute the contrastive loss L_{closs} as

$$L_{closs} = \frac{1}{N_w}\sum_{i=1}^{N_w} - \log \frac{\sum_{j=1}^{N_1} P_T/N_1}{\sum_{j=1}^{N_1} N_T/N_1} + a \tag{5.4}$$

with a being a constant set to 1; N_1 being set to the 20% of the total N^2; N_w being the number of windows. Accordingly, the total loss function of a depth prediction network employing window-based contrastive learning is defined as:

$$L_{total} = L_{depth} + w * L_{closs} \tag{5.5}$$

with L_{depth} the original depth loss from the depth prediction network, and w a weighting term for the contrastive loss.

Our WCL module is specifically designed for depth prediction tasks and can be seamlessly integrated into many networks. As shown in Fig. 5.6, the WCL block works on feature maps between two layers. This way, the module enables the contrast between the positive and negative pairs in the representation space.

5.3 Experimental Results

In this section, we evaluate our method on three main depth prediction tasks, i.e., depth completion, monocular depth estimation, and self-supervised monocular depth estimation. We provide an exhaustive evaluation of our proposal on KITTI dataset, NYU depth v2 dataset, and their splits. We select a set of models representative of

Method	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
MSG-CHN	820.145	223.987	**2.425**	0.979
MSG-CHN+WCL	**810.273**	**222.719**	2.428	**0.973**

Tab. 5.1: **Quantitative results on KITTI depth completion dataset [7].** Comparison between MSG-CHN and its WCL counterpart.

the three specific tasks, to which we apply our method to improve their accuracy consistently. Any model and its WCL variant are trained using the standard hyper-parameters reported in the original papers. We retrain them both with and without our module, allowing for a fair comparison under the same experimental setting (i.e., data, hardware support). This comes with little effort since our WCL module is a plug-and-play component easily embeddable in any depth prediction architecture. All the experiments are conducted using the PyTorch framework, on a single NVIDIA RTX 3090 GPU.

5.3.1 Depth completion

We consider two depth completion methods [10, 12], respectively MSG-CHN and Sparse-to-Dense. MSG-CHN [12] is a multi-branch guided cascade hourglass network for depth completion. Sparse-to-Dense [10] is built with an early-fusion network for multi-modal data. We insert our module between the last two layers of each branch in MSG-CHN and train it on the KITTI dataset [7]. In Sparse-to-Dense, we use ResNet18 [143] as the backbone to extract features, we insert our module between the last two layers of the decoder and train it on NYU Depth v2[2]. We use the same training parameters, except for the batch size, for both networks. The batch size is set to 8 and 12, respectively. For the two networks, we set $w = 0.2$, $N = 7$ and $w = 0.1$, $N = 7$ respectively. Tab. 5.1 and Tab. 5.2 show that both MSG-CHN and Sparse-to-Dense can benefit from our WCL module. Fig. 5.7 shows a qualitative comparison between MSG-CHN and its counterpart using WCL on a sample from the KITTI dataset. We can notice how our module allows for more precise boundaries at depth discontinuities. Fig. 5.8 instead compares Sparse-to-Dense models on the NYUv2 dataset, highlighting the same behavior observed for MSG-CHN.

Method	RMSE (mm)	REL	$\delta_{1.25^1}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$
Sparse-to-Dense	0.1097	0.0185	99.39	99.91	99.98
Sparse-to-Dense +WCL	**0.1038**	**0.0149**	**99.41**	**99.92**	**99.99**

Tab. 5.2: **Quantitative results on NYU Depth v2 Dataset [2].** Comparison between Sparse-to-Dense and its WCL counterpart.

Fig. 5.7: **Qualitative comparison on the KITTI depth completion dataset [1].** From top to bottom: RGB image, results of MSG-CHN [12], and MSG-CHN [12]+WCL, respectively. We zoom within the red line regions at the right; WCL achieves more precise object boundaries where the red arrow points.

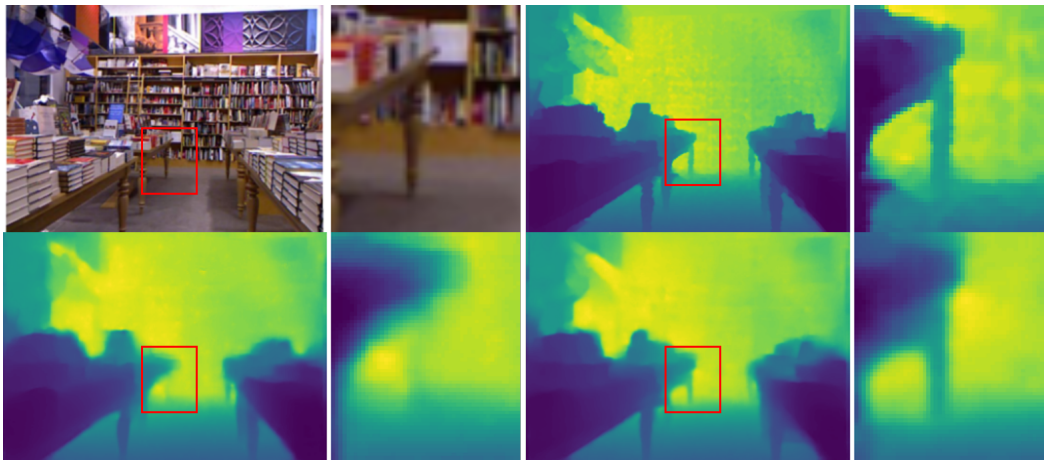


Fig. 5.8: **Qualitative comparison on the NYUv2 depth dataset[2].** From left to right, top to bottom, RGB image, ground truth, results of Sparse-to-Dense [10] and Sparse-to-Dense [10]+WCL. On the right of each image, we zoom into the red rectangle. With WCL, predicted depth maps expose more precise structures and boundaries.

Methods	*higher is better*			*lower is better*				
	$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$	AbsRel	Sq Rel	RMSE	RMSE *log*	log10
BTS-ResNet50	0.865	0.975	0.993	0.119	0.075	0.419	0.152	0.051
BTS-ResNet50+WCL	**0.871**	**0.977**	**0.994**	**0.117**	**0.072**	**0.409**	**0.149**	**0.050**
BTS-DenseNet-121	0.865	0.976	**0.995**	0.120	0.075	0.421	0.152	0.051
BTS-DenseNet-121+WCL	**0.869**	**0.977**	0.994	**0.117**	**0.072**	**0.413**	**0.149**	**0.050**

Tab. 5.3: **Quantitative results on NYUv2** [2] **– Monocular Depth estimation.** Comparison between BTS variants and their WCL counterparts.

5.3.2 Monocular Depth estimation

For this task, we select BTS [50] as a baseline. It is a state-of-the-art method using local planar guidance layers as geometric constraints to guide the features to depth upsampling in the decoding phase. We use BTS variants with different backbones, i.e., ResNet50 [143] and DenseNet-121 [161], on NYUv2 for the monocular depth estimation task. We use the same training parameters suggested by the authors [50], except for the batch size that is set to 10. For this task, we set $w = 0.1, N = 7$ for both the backbones. Tab. 5.3 shows that our WCL module allows consistent improvements over both BTS variants.

5.3.3 Self-Supervised Monocular Depth Estimation

Self-supervised monocular depth estimation eliminates the need for ground truth depth labels, which are usually hard to source. Supervision can be obtained in the form of monocular videos or stereo images. We select MonoDepth2 [57] as a state-of-the-art baseline for this task, simultaneously learning for depth and relative poses between consecutive frames in a video to implement the aforementioned self-supervised training scheme. We test it on the KITTI dataset [1] using the Eigen split [33]. Its training is realized by minimizing the photometric re-projection errors, either between temporally adjacent frames or stereo images. We use ResNet18 [143] as the backbone, process images resized to 192×640, and keep the same training parameters detailed by the authors [57]. In both cases, we evaluate our module setting $w = 0.01, N = 7$. Tab. 5.4 confirms that our method can also boost the

accuracy of self-supervised depth estimation frameworks.

Methods	Train	*higher is better*			*lower is better*			
		$\delta_{1.25}$	$\delta_{1.25^2}$	$\delta_{1.25^3}$	AbsRel	Sq Rel	RMSE	RMSE *log*
Monodepth2	M	0.871	0.957	0.980	0.118	0.912	4.911	0.196
Monodepth2 + WCL	M	**0.873**	**0.959**	**0.981**	**0.116**	**0.852**	**4.837**	**0.194**
Monodepth2	S	0.865	0.949	**0.975**	**0.109**	0.909	5.015	0.208
Monodepth2 + WCL	S	**0.867**	**0.951**	**0.975**	**0.109**	**0.892**	**4.961**	**0.207**

Tab. 5.4: **Quantitative results on KITTI Eigen split [33] – Self-Supervised Monocular depth estimation.** All methods are trained and tested with 192×640 images. The best results in each category are in **bold**; M: Monocular supervision; S: Stereo supervision.

5.3.4 Ablation Study

In this section, we conduct an ablation study to verify the impact of the different hyper-parameters of our WCL module. For the experiments in this section, we focus on the depth completion task; we use a subset of 10,000 samples from the KITTI depth completion dataset for training and evaluate the performance on the validation split. Images are center-cropped to 1216×256, to focus on regions with available LiDAR points. We use Sparse-to-Dense [10] as the baseline network and adopt ResNet-18 as the backbone. All the parameters are optimized using Adam ($\beta_1 = 0.9$, $\beta_2 = 0.99$), and the weight decay factor is set to 0.0001. The learning rate is initialized to 0.001, decayed by {0.5, 0.2, 0.1, 0.01} at epoch {10, 15, 20, 25}. The network is trained for 30 epochs using a batch size of 10 samples. RMSE, MAE, and iMAE are used as the evaluation metrics.

Window size. We measure how the window size over which the contrastive loss is applied impacts the final results. Tab. 5.5 collects the outcome of this experiment. We evaluate our module with window sizes, *3, 5, 7, 9, 11, 13, 15*. w is fixed instead, to *0.1*. We can observe that window with $N = 7$ outperforms others according to the main evaluation metric, RMSE. In contrast, bigger window sizes cannot improve further while increasing the computational requirements. A small window size ($N = 3$) yields negligible improvements, probably because most 3×3 regions are not

significant enough for applying contrastive learning effectively.

Window size (N)	RMSE (M)	MAE (mm)	iMAE (1/km)
baseline	930.326	269.866	2.575
3	927.818	266.248	2.536
5	925.169	267.290	2.813
7	**922.512**	264.044	2.030
9	925.358	267.614	2.045
11	926.511	**263.053**	2.639
13	925.072	265.628	2.429
15	925.074	264.810	**2.022**

Tab. 5.5: **Ablation results on the different window sizes on KITTI depth completion validation set.**

Shifted windows When using WCL, all windows are non-overlapped. Thus, distribution optimization occurs locally. Previous works exploiting windows processing as well [42, 112] use effective shifted window partitioning to introduce connections between neighboring non-overlapping windows. We ran an ablation study about whether shifted window partitioning can bring improvement in our method or not. Following [42, 112], we shift the windows by $(\frac{N}{2}, \frac{N}{2})$ pixels in the feature map and calculate the contrastive loss after computing the loss of the previous windows. We set $w = 0.1$ and $N = 7$. We shift the windows *1, 2, 3, 4* times. From the results in Tab. 5.6, we can conclude that shifting the windows does not bring improvement while increasing computational cost.

Embedded module Location Our WCL module can be easily embedded in between network layers, allowing to contrast pixels in the representation space. The baseline network has six layers in the decoder stage. We define them as *layer5, layer4, layer3, layer2, layer1,* and *layer0* from bottleneck to final layer. We performed an ablation study to determine how much improvement our module can bring when placed at different locations within the network. We set $w = 0.1$ and $N = 7$ in all the experiments except the last one. In the last one, we set $w_1 = 0.1$ for the contrastive loss between *layer1* and *layer0* and $w_2 = 0.0001$ between *layer2* and *layer1*. From the results in Tab. 5.7, we can find that the closer to the final layer, the better the

Shift number (N)	RMSE (M)	MAE (mm)	iMAE (1/km)
baseline	930.326	269.866	2.575
0	**922.512**	**264.044**	**2.030**
1	923.461	266.447	2.692
2	926.473	265.348	2.228
3	929.037	266.340	2.083
4	927.404	264.493	2.324

Tab. 5.6: **Ablation results on the shift number on the KITTI depth completion validation set.**

results. The outputs from the layers near the final layer present features strongly related to final depth maps, while the outputs near the bottleneck show higher-level, semantical information. Partitioning and contrasting the outputs near the bottleneck break semantic information and cause degradation.

Location	RMSE (M)	MAE (mm)	iMAE (1/km)
baseline	930.326	269.866	2.575
-1	922.512	264.044	2.030
-2	928.279	267.821	2.150
-3	929.924	264.688	2.594
-4	932.127	270.720	**1.992**
-5	940.022	270.118	2.120
-1 & -2	**912.286**	**263.886**	6.604

Tab. 5.7: **Ablation results on the different locations on KITTI depth completion validation set.** -1 denotes between *layer1* and *layer0*; -2 denotes between *layer2* and *layer1*; -3 denotes between *layer3* and *layer2*; -4 denotes between *layer4* and *layer3*; -5 denotes between *layer5* and *layer4*.

5.4 Conclusion

In this work, we presented a Window-based Contrastive Learning (WCL) module for depth prediction. Our approach partitions the image into windows, and the

contrastive loss is implemented within each. Accordingly, it constructs and sorts positive and negative pairs, then enlarges the gap between the two in feature space, which makes depth distribution more meaningful of the real depth in the scene. We evaluate our method on multiple depth prediction tasks, such as depth completion, depth estimation, and self-supervised depth estimation, reporting consistent improvements.

Conclusion

Summary of Thesis Achievements

In this thesis, several techniques leveraging deep learning have been contributed to propose compact and effective depth prediction models from both images and multi-modal data. Specifically, compact depth prediction models have been proposed for multi-modal data, which combine both image and LiDAR data. These models are lightweight and have very few parameters while achieving state-of-the-art performance according to the standard evaluation protocols that cater to resource-constrained devices, such as in-vehicle and edge ones. By exploiting inherent characteristics of depth distribution, contrastive learning methods, have been successfully deployed to enhance depth prediction models' learning ability to get more accurate results without increasing resource consumption.

Ongoing and future work

The ongoing work aims to introduce the affinity of image pixels in the depth prediction task. The widely used datasets, such as KITTI dataset [1] and Nuscenes [162], have semi-dense depth labels, therefore it is hard to get fine and rich-detailed depth results in supervised methods. Whether more supervisory information be introduced to improve prediction models' performance, especially in the results of object details? Images have rich texture information and it is easier to obtain semantic and detailed cues which can be helpful and serve as supplementary supervision can help improve the depth prediction networks. The current work focuses on using affinity from image features as additional supervision to improve the prediction accuracy of the model.

Future research endeavors will explore more deep learning methodologies that aim to advance depth prediction. Future research directions will include the study of other tasks (e.g. 3D reconstruction, binocular stereo depth, multi-view stereo, and others) using sparse accurate measurements. Incorporating sparse yet accurate depth measurements from either active depth sensors or conventional passive depth sensing methods into images can significantly enhance prediction accuracy while simultaneously eliminating scale ambiguity.

Bibliography

[1] A. Geiger, P. Lenz, and R. Urtasun. “Are we ready for autonomous driving? the kitti vision benchmark suite”. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3354–3361 (cit. on pp. 2, 12, 26, 42, 51, 58, 59, 66).

[2] N. Silberman et al. “Indoor segmentation and support inference from rgbd images”. In: *European conference on computer vision*. Springer. 2012, pp. 746–760 (cit. on pp. 2, 13, 14, 57–59).

[3] S. Hawe, M. Kleinsteuber, and K. Diepold. “Dense disparity maps from sparse disparity measurements”. In: *2011 International Conference on Computer Vision*. IEEE. 2011, pp. 2126–2133 (cit. on p. 4).

[4] L.-K. Liu, S. H. Chan, and T. Q. Nguyen. “Depth reconstruction from sparse samples: Representation, algorithm, and sampling”. In: *IEEE Transactions on Image Processing* 24.6 (2015), pp. 1983–1996 (cit. on p. 4).

[5] F. Ma et al. “Sparse sensing for resource-constrained depth reconstruction”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 96–103 (cit. on p. 4).

[6] J. Ku, A. Harakeh, and S. L. Waslander. “In defense of classical image processing: Fast depth completion on the cpu”. In: *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE. 2018, pp. 16–22 (cit. on p. 4).

[7] J. Uhrig et al. “Sparsity invariant cnns”. In: *2017 international conference on 3D Vision (3DV)*. IEEE. 2017, pp. 11–20 (cit. on pp. 4, 6, 12, 14, 26, 34, 35, 37, 42, 47, 57).

[8] Z. Huang et al. “Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion”. In: *IEEE Transactions on Image Processing* 29 (2019), pp. 3429–3441 (cit. on p. 4).

[9] F. Ma and S. Karaman. “Sparse-to-dense: Depth prediction from sparse depth samples and a single image”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 4796–4803 (cit. on pp. 4, 24, 28).

[10] F. Ma, G. V. Cavalheiro, and S. Karaman. “Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3288–3295 (cit. on pp. 4, 6, 28, 29, 34, 35, 44, 57, 58, 60).

[11] R. Fan et al. “A Cascade Dense Connection Fusion Network for Depth Completion”. In: *The 33rd British Machine Vision Conference*. 2022 (cit. on pp. 4, 20, 36–38, 51).

[12] A. Li et al. “A multi-scale guided cascade hourglass network for depth completion”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 32–40 (cit. on pp. 4, 20, 22, 24, 26–29, 35–38, 57, 58).

[13] J. Qiu et al. “Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3313–3322 (cit. on pp. 4, 27–29, 35–38).

[14] W. Van Gansbeke et al. “Sparse and noisy lidar completion with rgb guidance and uncertainty”. In: *2019 16th international conference on machine vision applications (MVA)*. IEEE. 2019, pp. 1–6 (cit. on pp. 4, 7, 20, 22, 28, 29).

[15] Y. Ke et al. “MDANet: Multi-Modal Deep Aggregation Network for Depth Completion”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 4288–4294 (cit. on pp. 4, 22, 24, 27–29, 36–38).

[16] S. Liu et al. “Learning affinity via spatial propagation networks”. In: *Advances in Neural Information Processing Systems* 30 (2017) (cit. on p. 4).

[17] X. Cheng et al. “Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 10615–10622 (cit. on pp. 4, 5, 27–29, 35).

[18] X. Cheng, P. Wang, and R. Yang. “Depth estimation via affinity learned with convolutional spatial propagation network”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 103–119 (cit. on pp. 4, 5, 24, 27–29).

[19] J. Park et al. “Non-local spatial propagation network for depth completion”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 120–136 (cit. on pp. 4, 5, 24, 27–29, 34).

[20] Y. Lin et al. “Dynamic spatial propagation network for depth completion”. In: *arXiv preprint arXiv:2202.09769* (2022) (cit. on pp. 4, 5, 34).

[21] A. Conti, M. Poggi, and S. Mattoccia. “Sparsity Agnostic Depth Completion”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2023, pp. 5871–5880 (cit. on pp. 4, 5).

[22] K. Rho, J. Ha, and Y. Kim. “GuideFormer: Transformers for Image Guided Depth Completion”. In: *CVPR*. 2022, pp. 6250–6259 (cit. on p. 4).

[23] Y. Zhang et al. “CompletionFormer: Depth Completion with Convolutions and Vision Transformers”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. CVPR. 2023 (cit. on pp. 4, 34).

[24] S. Zhao et al. “Adaptive context-aware multi-modal network for depth completion”. In: *IEEE Transactions on Image Processing* 30 (2021), pp. 5264–5276 (cit. on pp. 4, 29).

[25] H. Chen, H. Yang, and Y. Zhang. “Depth completion using geometry-aware embedding”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 8680–8686 (cit. on pp. 4, 35).

[26] X. Xiong et al. “Sparse-to-dense depth completion revisited: Sampling strategy and graph construction”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 682–699 (cit. on p. 4).

[27] Y. Wang et al. “LRRU: Long-short Range Recurrent Updating Networks for Depth Completion”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 9422–9432 (cit. on p. 5).

[28] K. Lai et al. “A large-scale hierarchical multi-view rgb-d object dataset”. In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 1817–1824 (cit. on p. 5).

[29] A. Saxena, S. Chung, and A. Ng. “Learning depth from single monocular images”. In: *Advances in neural information processing systems* 18 (2005) (cit. on p. 5).

[30] K. Karsch, C. Liu, and S. B. Kang. “Depth extraction from video using non-parametric sampling”. In: *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*. Springer. 2012, pp. 775–788 (cit. on p. 5).

[31] K. Karsch, C. Liu, and S. B. Kang. “Depth transfer: Depth extraction from video using non-parametric sampling”. In: *IEEE transactions on pattern analysis and machine intelligence* 36.11 (2014), pp. 2144–2158 (cit. on p. 5).

[32] J. Konrad, M. Wang, and P. Ishwar. “2d-to-3d image conversion by learning depth from examples”. In: *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2012, pp. 16–22 (cit. on p. 5).

[33] D. Eigen, C. Puhrsch, and R. Fergus. “Depth map prediction from a single image using a multi-scale deep network”. In: *Advances in neural information processing systems* 27 (2014) (cit. on pp. 5, 13, 35, 40, 59, 60).

[34] D. Eigen and R. Fergus. “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2650–2658 (cit. on pp. 5, 42).

[35] F. Liu, C. Shen, and G. Lin. “Deep convolutional neural fields for depth estimation from a single image”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 5162–5170 (cit. on p. 5).

[36] I. Laina et al. “Deeper depth prediction with fully convolutional residual networks”. In: *2016 Fourth international conference on 3D vision (3DV)*. IEEE. 2016, pp. 239–248 (cit. on pp. 5, 35).

[37] S. Aich et al. “Bidirectional attention network for monocular depth estimation”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 11746–11752 (cit. on p. 5).

[38] S. Lee et al. “Patch-wise attention network for monocular depth estimation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 3. 2021, pp. 1873–1881 (cit. on p. 5).

[39] A. Agarwal and C. Arora. “Attention Attention Everywhere: Monocular Depth Prediction with Skip Attention”. In: *arXiv preprint arXiv:2210.09071* (2022) (cit. on p. 5).

[40] C. Shu et al. “SideRT: A real-time pure transformer architecture for single image depth estimation”. In: *arXiv preprint arXiv:2204.13892* (2022) (cit. on p. 5).

[41] Z. Li et al. “DepthFormer: Exploiting Long-Range Correlation and Local Information for Accurate Monocular Depth Estimation”. In: *arXiv preprint arXiv:2203.14211* (2022) (cit. on p. 5).

[42] W. Yuan et al. “New crfs: Neural window fully-connected crfs for monocular depth estimation”. In: *arXiv preprint arXiv:2203.01502* (2022) (cit. on pp. 5, 8, 54, 61).

[43] A. Agarwal and C. Arora. “Attention attention everywhere: Monocular depth prediction with skip attention”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 5861–5870 (cit. on p. 5).

[44] K. Shim et al. “Depth-Relative Self Attention for Monocular Depth Estimation”. In: *arXiv preprint arXiv:2304.12849* (2023) (cit. on p. 5).

[45] F. Sheng et al. “Monocular Depth Distribution Alignment with Low Computation”. In: *arXiv preprint arXiv:2203.04538* (2022) (cit. on p. 5).

[46] Z. Li et al. “BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation”. In: *arXiv preprint arXiv:2204.00987* (2022) (cit. on p. 5).

[47] S. F. Bhat, I. Alhashim, and P. Wonka. “Adabins: Depth estimation using adaptive bins”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4009–4018 (cit. on p. 5).

[48] S. Shao et al. “IEBins: Iterative Elastic Bins for Monocular Depth Estimation”. In: *arXiv preprint arXiv:2309.14137* (2023) (cit. on pp. 5, 34).

[49] W. Yin et al. “Enforcing geometric constraints of virtual normal for depth prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5684–5693 (cit. on p. 5).

[50] J. H. Lee et al. “From big to small: Multi-scale local planar guidance for monocular depth estimation”. In: *arXiv preprint arXiv:1907.10326* (2019) (cit. on pp. 5, 43, 59).

[51] L. Liebel and M. Körner. “Multidepth: Single-image depth estimation via multi-task regression and classification”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 1440–1447 (cit. on p. 5).

[52] S. Qiao et al. “Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3997–4008 (cit. on pp. 5, 35).

[53] Z. Zhang et al. “Pattern-affinitive propagation across depth, surface normal and semantic segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 4106–4115 (cit. on p. 5).

[54] S. Shao et al. “NDDepth: Normal-Distance Assisted Monocular Depth Estimation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 7931–7940 (cit. on pp. 5, 34).

[55] R. Garg et al. “Unsupervised cnn for single view depth estimation: Geometry to the rescue”. In: *European conference on computer vision*. Springer. 2016, pp. 740–756 (cit. on pp. 5, 39).

[56] T. Zhou et al. “Unsupervised learning of depth and ego-motion from video”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1851–1858 (cit. on pp. 5, 6, 13, 35, 36, 43).

[57] C. Godard et al. “Digging into self-supervised monocular depth estimation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3828–3838 (cit. on pp. 6, 13, 35, 37, 39, 40, 43, 59).

[58] V. Guizilini et al. “3d packing for self-supervised monocular depth estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2485–2494 (cit. on pp. 6, 35, 43).

[59] C. Shu et al. “Feature-metric loss for self-supervised learning of depth and egomotion”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 572–588 (cit. on p. 6).

[60] M. Klingner et al. “Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 582–600 (cit. on p. 6).

[61] H. Jung, E. Park, and S. Yoo. “Fine-grained semantics-aware representation enhancement for self-supervised monocular depth estimation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 12642–12652 (cit. on p. 6).

[62] M. Poggi et al. “Towards real-time unsupervised monocular depth estimation on cpu”. In: *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2018, pp. 5848–5854 (cit. on pp. 6, 7, 35).

[63] M. Poggi et al. “Real-Time Self-Supervised Monocular Depth Estimation Without GPU”. In: *IEEE Transactions on Intelligent Transportation Systems* (2022), pp. 1–12. DOI: 10.1109/TITS.2022.3157265 (cit. on pp. 6, 7, 35).

[64] C. Zhao et al. “MonoViT: Self-Supervised Monocular Depth Estimation with a Vision Transformer”. In: *International Conference on 3D Vision*. 2022 (cit. on p. 6).

[65] Y. Zou, Z. Luo, and J.-B. Huang. “Df-net: Unsupervised joint learning of depth and flow using cross-task consistency”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 36–53 (cit. on p. 6).

[66] A. Ranjan et al. “Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 12240–12249 (cit. on p. 6).

[67] R. Ranftl et al. “Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer”. In: *IEEE transactions on pattern analysis and machine intelligence* 44.3 (2020), pp. 1623–1637 (cit. on p. 6).

[68] A. Gordon et al. “Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 8977–8986 (cit. on p. 6).

[69] Y. Yang, A. Wong, and S. Soatto. “Dense depth posterior (ddp) from single image and sparse range”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3353–3362 (cit. on pp. 6, 34, 44).

[70] A. Wong et al. “Unsupervised depth completion from visual inertial odometry”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1899–1906 (cit. on pp. 6, 34, 44).

[71] J. Choi et al. “Selfdeco: Self-supervised monocular depth completion in challenging indoor environments”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 467–474 (cit. on pp. 6, 34, 35, 39, 44).

[72] X. Cheng et al. “Noise-aware unsupervised deep lidar-stereo fusion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6339–6348 (cit. on pp. 6, 35).

[73] Z. Feng et al. “Advancing self-supervised monocular depth learning with sparse liDAR”. In: *Conference on Robot Learning*. PMLR. 2022, pp. 685–694 (cit. on pp. 6, 34, 35, 39, 42–44).

[74] F. Bartoccioni et al. “LiDARTouch: Monocular metric depth estimation with a few-beam LiDAR”. In: *Computer Vision and Image Understanding* 227 (2023), p. 103601 (cit. on p. 6).

[75] A. G. Howard et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv preprint arXiv:1704.04861* (2017) (cit. on p. 6).

[76] A. Howard et al. “Searching for mobilenetv3”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 1314–1324 (cit. on p. 6).

[77] N. Ma et al. “Shufflenet v2: Practical guidelines for efficient cnn architecture design”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 116–131 (cit. on p. 6).

[78] C. Yu et al. “Bisenet: Bilateral segmentation network for real-time semantic segmentation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 325–341 (cit. on p. 6).

[79] C. Yu et al. “Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation”. In: *International Journal of Computer Vision* 129.11 (2021), pp. 3051–3068 (cit. on p. 6).

[80] V. Peluso et al. “Enabling energy-efficient unsupervised monocular depth estimation on armv7-based platforms”. In: *Design, Automation & Test in Europe Conference & Exhibition*. 2019, pp. 1703–1708 (cit. on p. 7).

[81] V. Peluso et al. “Monocular Depth Perception on Microcontrollers for Edge Applications”. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2021) (cit. on p. 7).

[82] A. Cipolletta et al. “Energy-Quality Scalable Monocular Depth Estimation on Low-Power CPUs”. In: *IEEE IoT Journal* (2021) (cit. on p. 7).

[83] H. Zhao et al. “Icnet for real-time semantic segmentation on high-resolution images”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 405–420 (cit. on p. 7).

[84] S. Lee et al. “Deep architecture with cross guidance between single image and sparse lidar data for depth completion”. In: *IEEE Access* 8 (2020), pp. 79801–79810 (cit. on pp. 7, 35).

[85] J. Tang et al. “Learning guided convolutional network for depth completion”. In: *IEEE Transactions on Image Processing* 30 (2020), pp. 1116–1129 (cit. on pp. 7, 27, 29, 36–38).

[86] X. Li et al. “Gated fully fusion for semantic segmentation”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 2020, pp. 11418–11425 (cit. on p. 7).

[87] T. Xiao et al. “Unified perceptual parsing for scene understanding”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 418–434 (cit. on p. 7).

[88] T.-Y. Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125 (cit. on p. 7).

[89] Z. Zhou et al. “Unet++: A nested u-net architecture for medical image segmentation”. In: *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11 (cit. on pp. 7, 20, 22).

[90] H. Li et al. “Dfanet: Deep feature aggregation for real-time semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9522–9531 (cit. on p. 7).

[91] R. Hadsell, S. Chopra, and Y. LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. IEEE. 2006, pp. 1735–1742 (cit. on p. 7).

[92] P. Khosla et al. “Supervised contrastive learning”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 18661–18673 (cit. on p. 7).

[93] Z. Wu et al. “Unsupervised feature learning via non-parametric instance discrimination”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 3733–3742 (cit. on p. 7).

[94] K. He et al. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738 (cit. on pp. 7, 52, 53).

[95] A. Dosovitskiy et al. “Discriminative unsupervised feature learning with convolutional neural networks”. In: *Advances in neural information processing systems* 27 (2014) (cit. on p. 7).

[96] T. Chen et al. “A simple framework for contrastive learning of visual representations”. In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607 (cit. on pp. 7, 52, 53).

[97] Z. Xie et al. “Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16684–16693 (cit. on pp. 7, 53).

[98] X. Wang et al. “Dense contrastive learning for self-supervised visual pretraining”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3024–3033 (cit. on pp. 7, 53).

[99] K. Chaitanya et al. “Contrastive learning of global and local features for medical image segmentation with limited annotations”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 12546–12558 (cit. on pp. 7, 53).

[100] T. Xiao et al. “Region similarity representation learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10539–10548 (cit. on p. 7).

[101] T.-W. Ke, J.-J. Hwang, and S. X. Yu. “Universal weakly supervised segmentation by pixel-to-segment contrastive learning”. In: *arXiv preprint arXiv:2105.00957* (2021) (cit. on pp. 7, 53).

[102] I. Alonso et al. “Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 8219–8228 (cit. on p. 8).

[103] B. Xia et al. “Efficient Non-Local Contrastive Attention for Image Super-Resolution”. In: *arXiv preprint arXiv:2201.03794* (2022) (cit. on p. 8).

[104] J. Zhang et al. “Blind image super-resolution via contrastive representation learning”. In: *arXiv preprint arXiv:2107.00708* (2021) (cit. on p. 8).

[105] Y. Shen et al. “DCL: Differential Contrastive Learning for Geometry-Aware Depth Synthesis”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4845–4852 (cit. on p. 8).

[106] A. Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 8).

[107] A. Saxena, M. Sun, and A. Y. Ng. “Make3d: Learning 3d scene structure from a single still image”. In: *IEEE transactions on pattern analysis and machine intelligence* 31.5 (2008), pp. 824–840 (cit. on p. 8).

[108] X. Wang et al. “A depth estimating method from a single image using FoE CRF”. In: *Multimedia Tools and Applications* 74.21 (2015), pp. 9491–9506 (cit. on p. 8).

[109] L.-C. Chen et al. “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062* (2014) (cit. on p. 8).

[110] A. Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020) (cit. on p. 8).

[111] W. Wang et al. “Pyramid vision transformer: A versatile backbone for dense prediction without convolutions”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 568–578 (cit. on p. 8).

[112] Z. Liu et al. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022 (cit. on pp. 8, 9, 52, 54, 61).

[113] X. Dong et al. “Cswin transformer: A general vision transformer backbone with cross-shaped windows”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 12124–12134 (cit. on pp. 8, 54).

[114] X. Wang et al. “Non-local neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7794–7803 (cit. on p. 8).

[115] Z. Huang et al. “Ccnet: Criss-cross attention for semantic segmentation”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 603–612 (cit. on p. 8).

[116] E. Xie et al. “SegFormer: Simple and efficient design for semantic segmentation with transformers”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12077–12090 (cit. on p. 9).

[117] A. Geiger et al. “Vision meets robotics: The kitti dataset”. In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237 (cit. on p. 12).

[118] D. Nazir et al. “SemAttNet: Towards Attention-based Semantic Aware Guided Depth Completion”. In: *arXiv preprint arXiv:2204.13635* (2022) (cit. on pp. 20, 28).

[119] L. Liu et al. “Fcfr-net: Feature fusion based coarse-to-fine residual learning for depth completion”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 2021, pp. 2136–2144 (cit. on p. 20).

[120] Y. Zhang and T. Funkhouser. “Deep depth completion of a single rgb-d image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 175–185 (cit. on pp. 20, 35).

[121] P. Liu et al. “PDR-Net: Progressive depth reconstruction network for color guided depth map super-resolution”. In: *Neurocomputing* 479 (2022), pp. 75–88 (cit. on p. 20).

[122] K. Sun et al. “Deep high-resolution representation learning for human pose estimation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5693–5703 (cit. on pp. 20, 22).

[123] Y. Xin et al. “Reverse densely connected feature pyramid network for object detection”. In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 530–545 (cit. on p. 22).

[124] L.-C. Chen et al. “Rethinking atrous convolution for semantic image segmentation”. In: *arXiv preprint arXiv:1706.05587* (2017) (cit. on p. 24).

[125] Y.-H. Wu et al. “EDN: Salient object detection via extremely-downsampled network”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 3125–3136 (cit. on p. 25).

[126] W. Qilong et al. “ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 (cit. on p. 25).

[127] L. Huynh et al. “Boosting monocular depth estimation with lightweight 3d point fusion”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 12767–12776 (cit. on p. 27).

[128] M. Hu. *PENet*. https://github.com/JUGGHM/PENet_ICRA2021. 2021 (cit. on p. 27).

[129] Y. Xu et al. “Depth completion from sparse lidar data with depth-normal constraints”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2811–2820 (cit. on pp. 27, 29).

[130] M. Hu et al. “Penet: Towards precise and efficient image guided depth completion”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13656–13662 (cit. on pp. 27–29, 36–38).

[131] Y. Chen et al. “Learning joint 2d-3d representations for depth completion”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 10023–10032 (cit. on p. 29).

[132] A. Eldesokey, M. Felsberg, and F. S. Khan. “Propagating confidences through cnns for sparse data regression”. In: *arXiv preprint arXiv:1805.11913* (2018) (cit. on p. 29).

[133] Y. Cheng et al. “Locality-sensitive deconvolution networks with gated fusion for rgb-d indoor semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3029–3037 (cit. on p. 28).

[134] R. Fan et al. “Lightweight Self-Supervised Depth Estimation with few-beams LiDAR Data”. In: *The 34rd British Machine Vision Conference*. 2023 (cit. on p. 34).

[135] Z. Yan et al. “RigNet: Repetitive image guided network for depth completion”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 214–230 (cit. on p. 34).

[136] Y. Wang et al. “Decomposed Guided Dynamic Filters for Efficient RGB-Guided Depth Completion”. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2023) (cit. on p. 34).

[137] W. Zhou et al. “BEV@ DC: Bird’s-Eye View Assisted Training for Depth Completion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 9233–9242 (cit. on p. 34).

[138] C. Liu et al. “Single Image Depth Prediction Made Better: A Multivariate Gaussian Take”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17346–17356 (cit. on p. 34).

[139] L. Piccinelli, C. Sakaridis, and F. Yu. “iDisc: Internal Discretization for Monocular Depth Estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 21477–21487 (cit. on p. 34).

[140] C. Liu et al. “Va-depthnet: A variational approach to single image depth prediction”. In: *arXiv preprint arXiv:2302.06556* (2023) (cit. on p. 34).

[141] V. Guizilini et al. “Robust semi-supervised monocular depth estimation with reprojected distances”. In: *Conference on robot learning*. PMLR. 2020, pp. 503–512 (cit. on pp. 34, 43).

[142] A. Wong and S. Soatto. “Unsupervised depth completion with calibrated backprojection layers”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 12747–12756 (cit. on pp. 35, 39).

[143] K. He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on pp. 37, 57, 59).

[144] A. Wong, S. Cicek, and S. Soatto. “Learning topology from synthetic data for unsupervised depth completion”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1495–1502 (cit. on p. 39).

[145] Z. Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612 (cit. on p. 40).

[146] H. Fu et al. “Deep ordinal regression network for monocular depth estimation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2002–2011 (cit. on pp. 40, 43).

[147] S. Saha et al. “Learning to relate depth and semantics for unsupervised domain adaptation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 8197–8207 (cit. on p. 40).

[148] Y. You et al. “Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving”. In: *arXiv preprint arXiv:1906.06310* (2019) (cit. on p. 42).

[149] A. Paszke et al. “Automatic Differentiation in PyTorch”. In: *NIPS 2017 Workshop on Autodiff*. Long Beach, California, USA, 2017 (cit. on p. 42).

[150] F. Tosi et al. “Learning monocular depth estimation infusing traditional stereo knowledge”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9799–9809 (cit. on p. 43).

[151] R. Li et al. “Undeepvo: Monocular visual odometry through unsupervised deep learning”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 7286–7291 (cit. on p. 43).

[152] C. Wang et al. “Learning depth from monocular videos using direct methods”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2022–2030 (cit. on p. 43).

[153] Z. Yang et al. “Lego: Learning edge with geometry all at once by watching videos”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 225–234 (cit. on p. 43).

[154] Y. Liao et al. “Parse geometry from a line: Monocular depth estimation with partial laser observation”. In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 5059–5066 (cit. on p. 44).

[155] R. McCraith et al. “Lifting 2d object locations to 3d by discounting lidar outliers across objects and views”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2411–2418 (cit. on p. 46).

[156] Y. Wang et al. “CU-Net: LiDAR depth-only completion with coupled U-Net”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 11476–11483 (cit. on p. 46).

[157] R. Fan, M. Poggi, and S. Mattoccia. “Contrastive Learning for Depth Prediction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 3225–3236 (cit. on p. 50).

[158] P. Bachman, R. D. Hjelm, and W. Buchwalter. “Learning representations by maximizing mutual information across views”. In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 52).

[159] W. Wang et al. “Exploring cross-image pixel contrast for semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 7303–7313 (cit. on p. 53).

[160] H. Hu, J. Cui, and L. Wang. “Region-aware contrastive learning for semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 16291–16301 (cit. on p. 53).

[161] G. Huang et al. “Densely connected convolutional networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4700–4708 (cit. on p. 59).

[162] H. Caesar et al. “nuscenes: A multimodal dataset for autonomous driving”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631 (cit. on p. 66).