Self-supervised Visual Geometry Learning

Yiran Zhong

Committee in charge:

Professor Hongdong Li, Primary and Chair
Professor Yuchao Dai
Reader Henry Gardner
Professor Nicholas Barnes
Professor Richard Hartley

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

December 2020

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

Yiran Zhong
31 December 2020

Thesis Outcome

Publications

1. **Yiran Zhong**, Yuchao Dai, Hongdong Li. Depth Completion using Piecewise Planar Model, In *arXiv preprint arXiv:2012.03195, 2020*

2. **Yiran Zhong**, Yuchao Dai, Hongdong Li. Efficient Depth Completion Using Learned Bases, In *arXiv preprint arXiv:2012.01110, 2020*

3. **Yiran Zhong***, Jianyuan Wang*,Yuchao Dai, Stan Birchfield, Kaihao Zhang, Nikolai Smolyanskiy, Hongdong Li. Deep Two-View Structure-from-Motion Revisited, In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn. (CVPR) 2021 submitted*.

4. Jianyuan Wang*, **Yiran Zhong***, Yuchao Dai, Kaihao Zhang, Pan Ji, Hongdong Li. Accurate Optical Flow Estimation by Learned Matching Cost, In *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS) 2020*.

5. Xuelian Cheng*, **Yiran Zhong***, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Hongdong Li, Tom Drummond, Zongyuan Ge. Hierarchical Neural Architecture Search for Deep Stereo Matching, In *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS) 2020*.

6. **Yiran Zhong**, Charles Loop, Wonmin Byeon, Stan Birchfield, Yuchao Dai, Kaihao Zhang, Alexey Kamenev, Thomas Breuel, Hongdong Li, Jan Kautz. Displacement Invariant Cost Computation for Efficient Stereo Matching, In *arXiv preprint arXiv:2012.00899, 2020*

7. **Yiran Zhong**, Pan Ji, Jianyuan Wang, Yuchao Dai, Hongdong Li. Unsupervised Deep Epipolar Flow for Stationary or Dynamic Scenes, In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn. (CVPR) June 2019.*

8. Xuelian Cheng*, **Yiran Zhong***, Yuchao Dai, Pan Ji, Hongdong Li. Noise-Aware Unsupervised Deep Lidar-Stereo Fusion, In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn. (CVPR) June 2019.*

9. **Yiran Zhong**, Hongdong Li, Yuchao Dai. Open-world stereo video matching with deep rnn. In *Proc. Eur. Conf. Comp. Vis. (ECCV) August 2018.*

10. **Yiran Zhong**, Yuchao Dai, Hongdong Li. Stereo Computation for a Single Mixture Image. In *Proc. Eur. Conf. Comp. Vis. (ECCV) August 2018.*

11. **Yiran Zhong**, Yuchao Dai, Hongdong Li. 3d geometry-aware semantic labeling of outdoor street scenes, In *Proc. IEEE Int. Conf. Patt. Recogn. (ICPR) 2018.*

12. **Yiran Zhong**, Yuchao Dai, Hongdong Li. Self-supervised learning for stereo matching with self-improving ability, In *arXiv preprint arXiv:1709.00930, 2017*

Patent

- **Yiran Zhong**, Charles Loop, Wonmin Byeon, Stan Birchfield, NEURAL NETWORK SYSTEM FOR STEREO IMAGE MATCHING. *US Patent Application, Filed June 2020*

To my family for their love and support.

Acknowledgments

My PhD is a wonderful journey in my life. I am glad I have met an incredible group of people during the journey.

First, I would like to thank my primary supervisor, Hongdong Li, for his support throughout these years. Thanks to his insightful guidance, during my four years' PhD journey, I could barely feel confusions and depressions. I always think a supervisor to a PhD is like a captain to a sailor. A good captain should always have a big picture while a sailor only needs to follow commands. So I started with his ideas. After I started to understand the intuition behind, he revealed the big picture and let me work toward it. It avoided the depression I may have when I got stumbled in research.

I would also like to acknowledge my supervisor Yuchao Dai. In my mind, he is more like a friend to me rather than a supervisor. When he was in ANU, I often came to his office to discuss some new ideas and analyze their feasibility. We argued a lot, but it largely sped up my research progress so that I did not need to try many errors before finding the correct path. Most of time people are unable to find defects in their own ideas but they can be easily picked up from others' view. When Yuchao was back to China, we continued this connection. I often call him in the middle of the night – most of my ideas are popped out that time, and I can always get instant answers. I was also privileged to be studying under the supervision of Richard Hartley. We discussed the evolution of stereo matching and self-supervised learning in the stereo matching problem.

Apart from my supervisors, I would like to acknowledge my close friend Pan Ji. We joined in the ANU almost at the same time, when he was a PhD student of Hongdong and I was a master student. Hongdong introduced him to me at the end of 2014. He wanted him to show me how to do research and how a PhD life was like. Before I met him, I had no idea what kind of ideas can become a paper, which experiment is needed to support the claim and how to write a good paper. Pan Ji taught me all these things nearly hand by hand and we published two papers together.

I am fortunate to have worked with my colleagues on the third floor of Brian Anderson Building including Laurent Kneip, Dingfu Zhou, Xibing Song, Jiaolong Yang, Xin Yu, Cristian Rodrigiuze, Yi Zhou, Gao Zhu, Zhipeng Xiao, Liu Liu, Liyuan Pan, Kaihao Zhang, Yao Lu, Jianyuan Wang, Hongguang Zhang, Shihao Jiang, Jing Zhang and Yujiao Shi. A special thank to Xuelian Cheng for helping me draw network architectures in my papers.

I also enjoyed my internship at NVIDIA. I appreciate Charles Loop, Nikolai Smolyanskiy, Stan Birchfield and Jan Kautz for inviting me to join NVIDIA and

hosting me in the Redmond office. I am glad that my internship projects in both the autonomous driving team and NVIDIA research team are aligned with my thesis topic. I have seen many outstanding demos from Nikolai's team and I was excited to work with these brilliant people including Ollin Boer Bohan, Alexey Kamenev, FangKai Yang, Wonmin Byeon, Thomas Breuel, Peng Fei, Xueqiang Wang, Zhuohao Zhang, Xiaoshu Liu, Jingguang Zhou and many others. I would also thank Stan Birchfield for revising my papers and making lots of valuable comments to my slides and helping me to make better presentations.

Last but not least, I would like to express my acknowledgement to my family. Without their love and supports I would not be able to accomplish my PhD. I dedicate this thesis to them.

Abstract

Visual geometry learning aims to recover 3D geometry information *i.e.,* surface normal, depth maps and camera poses from images. As a classic task in computer vision, this problem has been studied extensively for decades. It contains depth completion, stereo matching, monocular depth estimation, optical flow, visual odometry, structure from motion and *etc*. This thesis is dedicated to solving these problems from both conventional learning and deep learning perspectives.

Like most data-driven methods, supervised deep learning-based methods require a large amount of labeled training data and suffer limited generalization ability. Self-supervised learning is a technique that allows a network to learn feature representations without labeled data. In this thesis, we investigate the problem of applying self-supervised learning techniques to visual geometry learning and push the limit of the state of the art in terms of accuracy, speed, and generalization ability in visual geometry recovery tasks.

In the depth completion task, two conventional optimization-based methods are proposed. The first one assumes a dense depth map can be approximated by a weighted sum of a set of principal components and enforces this assumption as a global geometric constraint. A colour-guided auto-regression model is applied to make the estimated depth map have sharp object boundaries. The proposed method can be efficiently solved in a closed form and outperforms previous methods. The other method further enforces a piecewise planar model to depth completion task and formulates it as a continuous Conditional Random Field (CRF) optimization problem. Experiments show that the proposed method is faster and more accurate than previous methods.

In the stereo matching task, we propose to solve this problem through a deep self-supervised framework. Conventional optimization-based methods often require several seconds to minutes to process a sample, which makes them infeasible for time-critical applications such as autonomous driving and robotics. Moreover, supervised deep methods often require a large number of ground truth labels for training and suffer limited generalization capability. By leveraging self-supervised learning, our self-supervised stereo matching networks will not need any labeled data and can adapt themselves to new scenarios on-the-fly. The key idea is to make several assumptions of scenes and formulate them into loss functions, then optimize them through backpropagation. The loss functions are similar to the energy functions in conventional optimization-based methods but we are allowed to use more complex loss functions to describe a scene more precisely. Experiments demonstrate that the proposed methods have better performance in terms of both speed and accuracy. A similar strategy is also applied to the LiDAR-Stereo fusion task. A "feedback loop" is proposed to deal with the noise in LiDAR measurements. We also extend

stereo matching to stereo video matching problem by utilizing convolutional LSTM modules to handle temporal consistency in videos. To deal with time-critical applications, we present a super-efficient stereo matching network structure that can process HD images at 100 FPS. We also leverage AutoML techniques *i.e.,* neural architecture search (NAS), to find an optimal architecture for deep stereo matching and achieve top 1 accuracy among various benchmarks with far less trainable parameters.

We further define a new problem called single mixture image depth estimation. Here, the single image can be a mixture of a stereo pair in a form of $I = \alpha I^{left} + (1 - \alpha) I^{right}$. Depending on the choice of α, this task can be seen as Red-Cyan depth, Double vision depth, and monocular depth estimation. Instead of brute force regressing depth from a single image, we divide the task into two sub-tasks: image separation and stereo matching. We first decouple the mixed image through an image separation module and then do stereo matching on the separated pairs. The whole system only needs original stereo pairs as supervisions and has better performance than previous methods.

In the optical flow task, we investigate multiple ways to enforce the global epipolar constraint in self-supervised optical flow estimation. For stationary scenes, a fundamental matrix constraint is present. We first estimate a fundamental matrix from matching points and regularize optical flow with the Sampson distance. For dynamic scenarios, we propose a low-rank constraint and a union-of-subspaces constraint. They avoid explicitly computing the fundamental matrix as well as multi-motion estimation. Experiments on various benchmarks demonstrate the effectiveness of our method.

In the structure from motion (SfM) task, we revisit existing deep learning-based approaches and find that they all formulate the problem in ways that are fundamentally ill-posed, relying on training data to overcome the inherent difficulties. In contrast, we propose a new deep framework that leverages the well-posedness of the classic SfM pipeline. We also propose a scale-invariant matching module to handle the scale ambiguities in the monocular SfM task. Our framework outperforms all state-of-the-art two-view SfM methods by a clear margin on various benchmarks in both relative pose estimation and depth estimation tasks.

Apart from these tasks, we also show how to apply geometry information to high-level computer vision tasks, *i.e.,* RGB-D semantic segmentation. We propose a natural and direct 3D representation to encode RGB-D data and regularize them with a light-weighted 3D convolutional network. State-of-the-art performance of our method on various datasets suggests that such a simple 3D representation is effective in incorporating 3D geometric information.

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction and Literature Overview

1.1 Introduction

3D geometry information is important for a wide range of applications such as autonomous driving, robotics, virtual and augmented reality, action recognition, face recognition, and so on. Recovering such information has become one of the most fundamental and important topics in computer vision. Although it has been extensively studied for several decades but is still far from a solved problem. 3D geometry information can be either extracted from camera images or measured by active ranging sensors (*e.g.,* RADAR, LiDAR, Time-of-Flight (ToF) camera, and structured-light sensor). The former can be traced back to the cooperative computation of stereo matching in 1976 [1] and the latter begins with structured light in the early 1970s [2]. The fusion of cameras and ranging sensors are also of high interest [3].

Visual 3D geometry information recovery refers to estimating dense depth maps and motions of a scene from the data acquired by a vision system. Five typical visual 3D geometry recovery problems exist in computer vision: i) *depth completion*, ii) *stereo matching*, iii) *single image depth estimation*, iv) *optical flow*, and v) *structure from motion*. *Depth completion* is the problem of interpolating a set of sparse depth points in a scene. It often takes a sparse depth map and a reference image as inputs. The sparse depth map is generated by projecting the set of sparse depth points to the reference image plane. The goal of this task is to generate a dense depth map corresponding to the reference image. *Stereo matching* mimics human stereo vision and tries to recover a dense depth map from a pair of rectified stereo images. Since the depth is inversely proportional to the displacement (disparity) of each pixel, estimating the depth is, in theory, equivalent to estimating the disparity. *Single image depth estimation* aims to estimate a depth map from a single image. The single image here can be a monocular image, a red-cyan image, a double vision image, or other forms of mixture images. Given a mixture image, the problem is to estimate a dense depth map from the mixture image and unmix the image into two images. The *optical flow* task is to compute per-pixel movement on the image plane between two consecutive frames. It serves as an essential step for motion estimation. The task of *structure from motion*

is trying to recover both the camera motion and the dense depth map of each frame from a monocular video.

Back to early 2010s, researchers often solved these problems by graph-based energy minimization methods including graph cut [4], min-cut/max-flow [5] and exact optimization [6], or semi-global methods [7]. Beginning with the success of deep convolutional networks on the 2012 ImageNet Challenge [8], deep learning has overwhelmed the computer vision community with its ability to achieves top scores across different fields and benchmarks. By providing a large amount of training data, a deep network can be well trained through backpropagation. We refer to the former methods as traditional methods and the latter as deep methods. Traditional methods and deep methods both have their own advantages and drawbacks. In traditional methods, they first build a target energy function with two major components: a data term and a regularization term. The data term contains a condition that each pixel should satisfy while the regularization term constraints neighbouring pixels. Then they minimize the energy functions iteratively or in closed-form. The advantage of these methods is that they do not require ground truth labels, whereas the drawback often lies in the long processing time. On the other hand, as data-driven methods, deep methods still suffer from some well-known drawbacks such as requiring a large amount of labeled training data, and limited generalization ability to unseen domains.

To remedy this drawback, the concept of self-supervised learning is proposed. In 1994, Virginia *et al.* [9] first introduce the self-supervised learning in the image classification task. Instead of using human-annotated labels as supervision, they minimize the disagreement between the outputs of network processing patterns from different sensory modalities and achieve similar performance than supervised methods. In the deep learning era, we often leverage the self-supervised learning technique to extract features from large unlabeled data. It can be done by defining a series of pre-designed tasks where their labels can be automatically generated and train a network with these tasks. There are various of pre-designed tasks including colourizing grayscale image [10], image inpainting [11], image jigsaw puzzle [12], classifying corrupted images [13]. With these self-trained features, we can fine-tune them to a specific task with limited human-labeled data and achieve better performance.

Most visual geometry problems can be solved using self-supervised learning techniques as the training signals can be automatically generated by exploiting relations between input signals and prior knowledge of the output. For example, by assuming Lambertian surfaces, we can use image warping errors [14] to supervise the matching process in stereo matching [15], optical flow [16] and structure from motion. We can also add some priors of a scene to get better performance such as smoothness [14], left-right consistency [14] and epipolar constraints [16].

In this thesis, we leverage both traditional methods and deep methods to solve the above 3D geometry recovery tasks. Specifically, we use traditional methods to solve the depth completion problem and *self-supervised* deep learning-based methods to solve the rest. Through this technique, our methods can self-adapt to different unseen scenarios in an online tuning scheme.

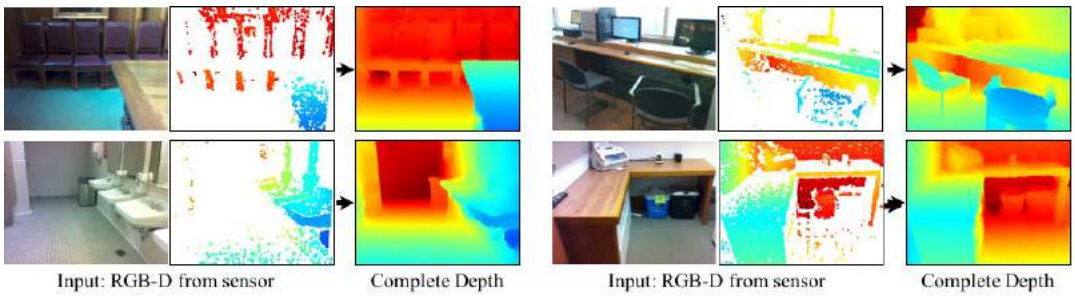


Figure 1.1: [1]**Illustration of depth completion.** Given a sparse depth map and its reference image, depth completion task is to fill the missing depth values to achieve a dense depth map.

1.2 Visual Geometry Recovery

Visual geometry recovery aims to extract 3D geometry information such as depth maps and motions from the data obtained by cameras or range sensors. Five visual geometry estimation problems are considered in this thesis: 1) *depth completion*, 2) *depth from a stereo camera*, 3) *depth from a single image*, 4) *image motion from a monocular camera* and 5) *depth and camera motion from a monocular video*. We will introduce these problems as follows.

1.2.1 Depth Completion

Given an incomplete depth map, as illustrated in Figure 1.1, depth completion task is to fill the missing depth values to get a dense depth map. A colour image that is captured with a camera is often used as the guided image. Since most range sensors can only achieve sparse or semi-dense depth maps, completing missing depth values can increase the quality of depth maps for better applicability. This task has drawn increasing attention recently due to the rising interest of autonomous driving, augmented/virtual reality and robotics. Although this task can be partially addressed by traditional image in-painting techniques, extra knowledge such as depth smoothness, normal smoothness, colour guidance and *etc.* is yet to be utilized for achieving an accurate dense depth map.

Depth completion includes three sub-tasks: depth inpainting, depth super resolution and depth reconstruction from sparse samples. Anisotropic diffusion [18] is a popular method for image inpainting, and it has been successfully adapted to the depth inpainting task in [19]. Energy Minimization based depth inpainting methods [20, 21] take characteristics of depth maps as a regularization term in energy minimization and generate more plausible results. Exemplar-based Filling [22] works well in image inpainting but creates extra challenges in depth inpainting due to the lacking of textures in depth maps. Matrix Completion [23] assumes that a depth map lies in a low-dimensional subspace and can be approximated by a low-rank

[1]Figures are from [17].

matrix. The main goal for depth super-resolution task is to increase the spatial resolution of a depth image to match a high resolution colour image. The low-resolution depth maps are completed in this case. By assuming depth discontinuities are often aligned with colour changes in the reference image, this task can be solved through Markov Random Fields (MRF) [24, 25]. Such methods can be easily adapted to depth reconstruction from sparse samples task as they do not require the depth points to be regularly sampled [26]. In order to intensively use the structural correlation between colour and depth pairs in colour guided depth super-resolution, non-local means (NLM) was introduced as a high-order term in regularization [27]. [28] swaps the Gaussian kernel in the standard NLM to a bilateral kernel to enhance the structural correlation in colour-depth pairs and proposed an adaptive colour-guided auto-regressive (AR) model that formulates the depth upsampling task as AR prediction error minimization, which owns a closed-form solution. A few approaches employ sparse signal representations for guided upsampling making use of the Wavelet domain [29], learned dictionaries [30] or co-sparse analysis models [31]. [32] and [33] leverage edge-aware image smoothing techniques and formulate it as a weighted least squares problem while [33] also applied coarse-to-fine strategy to deal with the very sparse situation. [34] proposed an efficient depth completion algorithm that achieves the state-of-the-art performance on KITTI depth completion benchmark [35] that only uses image processing operations.

Deep network is firstly introduced to the depth super-resolution task in early 2016. By treating a depth map as a gray-scale image, image super-resolution networks [36] can be directly applied [37]. Riegler *et al.* [38] proposed a deeper depth super-resolution network that has faster convergence and better performance. A natural barrier for applying deep methods to other depth completion tasks is the irregular depth pattern as standard convolution layers are designed for regular grid inputs. Sparsity Invariant CNNs [35] addresses this problem and tried to handle the sparse and irregular inputs by introducing invalid masks in convolution layers. On the other hand, [39] claims that standard convolution layers can handle sparse inputs of various densities without any additional mask input. In [40], it proves that leveraging semantic information can improve the depth completion performance. [17] utilizes a deep network to infer the surface normal and occlusion boundaries from colour images and inpaint raw depth maps through a global optimization. The Surface normal constraint is also applied to depth reconstruction from sparse samples task as well [41, 42].

1.2.2 Depth From A Stereo Camera

Similar to human, a machine can recover dense depth maps from a stereo camera. As shown in Figure 1.2, stereo matching aims to find matching points between a pair of rectified stereo images in horizontal scan-lines and generate a disparity map. Then a depth map can be easily recovered from the disparity map using the equation Bf/d, where B is the stereo baseline, f is the focal length and d is the disparity. Stereo matching has been extensively studied for decades in computer vision and numerous

methods have been proposed throughout the years.

In 1976, Marr *et al.* [1] proposed two physical constraints that stereo matching should satisfy: 1) uniqueness, and 2) continuity. Uniqueness means that for any points in an image, there should be at most one matched point in the other image. The continuity means disparities change slowly in most parts. More assumptions have been proposed later on, *e.g.*, brightness constancy constraint that matched points should have similar intensities, ordering that matched points should be in the same order in both views, and scanline search that the matched points should be on the same row of the stereo pairs. These assumptions have been leveraged in both local and global methods. Local methods [43, 44] often follow four consecutive steps: 1) compute costs at a given disparity for each pixel; 2) sum up the costs over a window; 3) select the disparity that has the minimal cost; 4) perform a series of post-processing steps to refine the final results. These methods gain speed but lose accuracy as they only consider local information. On the other hand, global methods [45, 46] treat the image as an undirected graph and try to assign each node with a disparity that minimize a global energy function. Depending on the energy function, global methods may achieve global optimal but are often slow in optimization. Semi-global matching (SGM) [7] is a compromise between these two extremes, which could achieve more accurate results than local methods without sacrificing speed significantly.

The first deep stereo matching algorithm is proposed in [47], which utilizes a deep network to learn better features and costs. One of the advantages of deep networks is that they can learn to deal with difficult scenarios, such as occlusions, repetitive textures or textureless regions, from ground truth data. Most deep stereo methods follow the traditional stereo matching pipeline and try to replace parts of them with a deep network. For example, Seki *et al.* [48] tries to learn a better regularization term, Khamis *et al.* [49] aims to learn a better refinement and Zhang *et al.* [50] proposed a better cost aggregation. Another trend of deep stereo matching is to allow a network to directly regress a disparity from inputs [51, 52, 53, 54] using an encoder-decoder structure. These methods often suffer generalization issues [15] and are vulnerable to adversary attacks [55]. Rather than direct regression, GCNet [56] and its followers [50, 14, 57] propose a volumetric structure that embeds the traditional stereo pipeline. It builds a 4D feature volume using features extracted from stereo pairs, and processes it with multiple 3D convolutions. The concept behind is semi-global matching. However, they suffer from high processing times and high GPU memory consumption.

1.2.3 Depth From A Single Image

Depth from a single image originally means monocular depth estimation, *i.e.*, inferring depth from a monocular image. In this thesis, we extend this concept to inferring depth from a single *mixture* image. The mixture image $\mathbf{I}$ is a composition of an original stereo image pair $\mathbf{I}_L$ and $\mathbf{I}_R$, *i.e.*, $\mathbf{I} = f(\mathbf{I}_L, \mathbf{I}_R))$, where $f(\cdot)$ represents different image composition operators to generate the mixture image. As illustrated

[2]Figures are from [58].

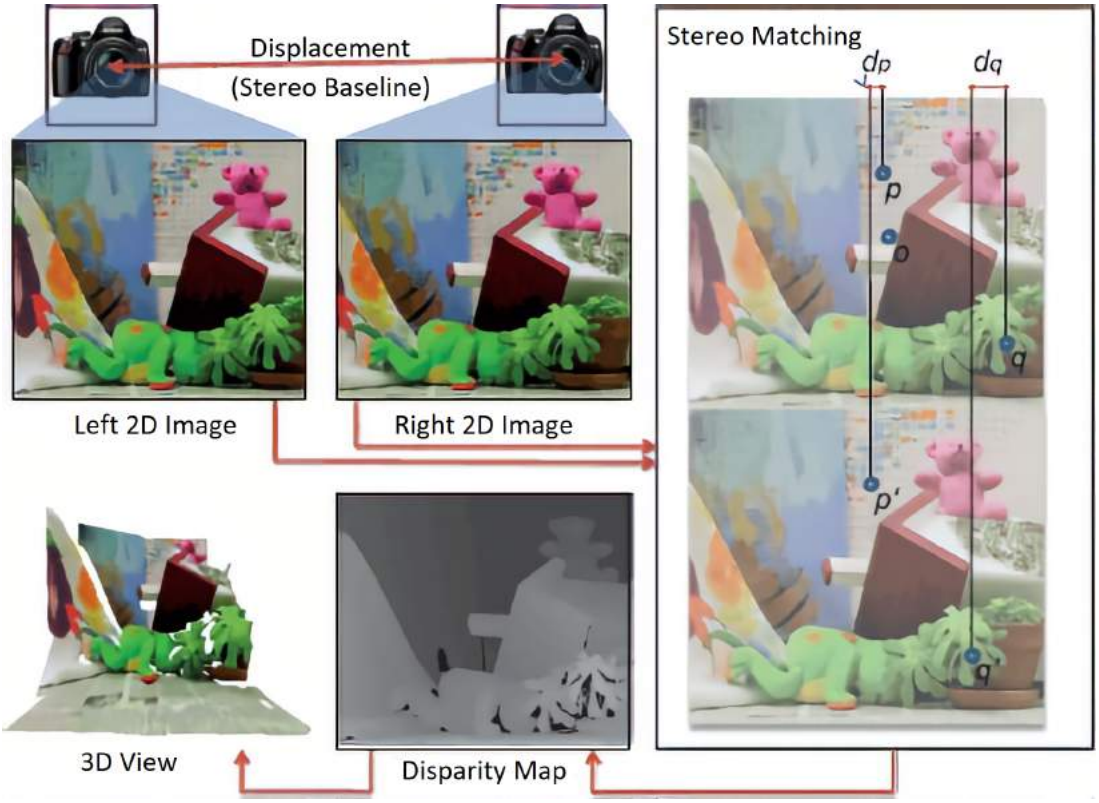


Figure 1.2: [2]**Illustration of stereo matching.** Given a pair of rectified stereo images, the stereo matching task is to find corresponding points in the horizontal scan-lines and generate a disparity map.

in Figure 1.3, the task is to recover the stereo pairs $\mathbf{I}_L, \mathbf{I}_R$ as well as the dense depth map simultaneously. We define the first sub-task as view recovery and the other one as depth recovery. The view recovery appears to be a blind signal separation (BSS) task, *i.e.,* separating an image into two different component images. Traditional BSS methods are not suitable for our task as they assume the two component images are statistically independent [59]. Such assumptions do not hold in our view recovery task. Similarly, other image layer separation methods such as reflection removal and highlight removal methods [60, 61] will not work well in our problem. Based on the image composition operators, our task can be treated as de-anaglyphy, double vision, view synthesis and monocular depth estimation, and so on.

Double vision is the simultaneous perception of two images (a stereo pair) of a single object in the form of a single mixture image. Specifically, a double vision image (*cf.* Figure 1.3) can be constructed by $I = f(I_L, I_R) = (I_L + I_R)/2$, *i.e.,* , the image composition is f a direct average of the left and the right images. It is a highly ill-posed problem as the two images are highly correlated. It can be seen as a constraint case of debluring task that needs to simultaneously recover two clean images as well as the encoded depth map. To the best of our knowledge, there are no previous methods that address the same problem. In this thesis, we propose

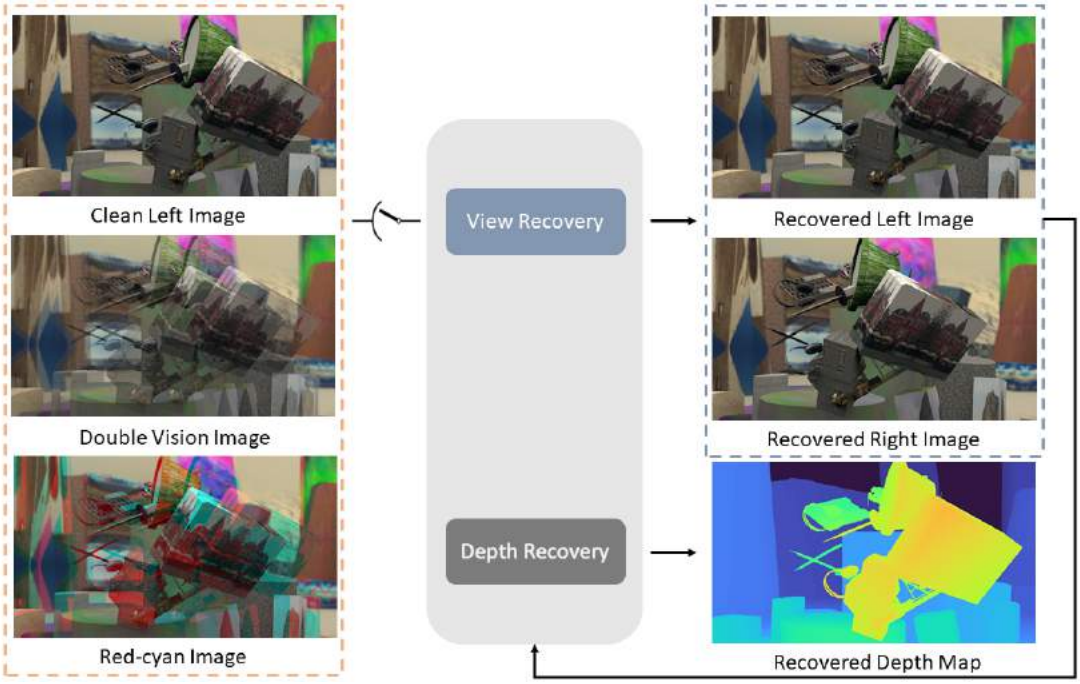


Figure 1.3: **Illustration of depth from single mixture image.** The task is to recover a dense depth map and restore full colour images from one type of mixture image.

a novel deep-learning based solution to this problem. Our network consists of an image separation module and a stereo matching module, where the two modules are optimized jointly. Under our framework, the solution of one module benefits the solution of the other.

De-anaglyphy is to restore full colour stereo pairs from a red-cyan image and estimate its disparity maps. An anaglyph (*cf.* Figure 1.3) is a single image created by selecting chromatically opposite colours (typically red and cyan) from a stereo pair. Thus given a stereo pair I_L, I_R, the image composition operator $f(\cdot)$ is defined as $I = f(I_L, I_R)$, where the red channel of I is extracted from the red channel of I_L while its green and blue channels are extracted from I_R. Traditionally it can be done by matching cross channel points between red-cyan images [62] or iteratively solving the problem for colour restoration and stereo computation [63].

View synthesis and monocular depth estimation is to infer an image captured from another view from the given image [64] and estimate a dense depth map [65]. In this case, the mixed image will be $I = f(I_L, I_R) = I_L$. Given the input I_L and the synthesised right image $\hat{I}_R$, we can compute a dense disparity map. By jointly optimizing these two tasks, we can achieve better synthesised images that obey geometry constraints.

Besides academic interest, this task also has wide applications. For de-anaglyphy, there are thousands of anaglyph videos where the original stereo images are not necessarily available. Recovering stereo images and their corresponding disparity maps will significantly improve the users' real 3D experience. For double vision, it can be

Figure 1.4: **Illustration of optical flow.** Optical flow is defined as a dense pixel-wise movement between two consecutive frames.

treated as a wide baseline deblur problem and we proved that the depth information can be recovered from blurred images. For monocular depth estimation, it provides a good prior depth information for Simultaneous Localization And Mapping (SLAM), semantic segmentation, bokeh effects, and so forth.

1.2.4 Image Motion From A Monocular Camera

We refer image motion as the apparent movements of pixels between two consecutive frames. In computer vision, it also refers to optical flow problem [66]. Figure 1.4 visualizes flow vectors and coloured flow fields in the UCL dataset [67], MPI Sintel dataset[68] and Middlebury Flow dataset [69]. Optical flow estimation is a fundamental problem in computer vision with many applications. Since Horn and Schunck's seminal work [66], various methods have been developed using variational optimization [70, 71, 72], energy minimization [73, 74, 75, 76], or deep learning [77, 78, 79, 80]. Although optical flow has been extensively studied for several decades, estimating dense and accurate flow field is still a challenging task, especially for large motions and textureless areas and thus remains an active topic in computer vision.

Optical flow and stereo matching are similar in the sense that they both aim to find the pixel correspondence between a pair of images. They share the same assumptions such as brightness constancy and continuity. The difference between them is that the searching space in optical flow increases from 1D to 2D compared to stereo matching. Depending on the way of enforcing the continuity constraint, traditional optical flow methods can be classified into local methods [81] and global methods [66].

Local methods compute flow vectors within a local patch independently. A well-known technique for local methods is PatchMatch [82]. It assumes that a large number of random guesses often contain good initials and neighboring pixels should have coherent matches, therefore we can propagate matching points information to their neighbors once a good match is found. PatchMatch is able to handle large displacements, and becomes popular in optical flow and stereo matching problems [83, 84].

Global methods consider the global smoothness of the whole image. They often formulate this problem as energy optimization that contains a data term and a global smoothness term. These methods attract more attention than local methods as they often have better accuracy, especially for textureless regions. Global methods can also cooperate with geometry constraints. By assuming the scene is mostly rigid, Valgaerts *et al.* [85] introduce a variational model to simultaneously estimate the fundamental matrix and the optical flow. Wedel *et al.* [86] utilize the fundamental matrix prior as a weak constraint in a variational framework. Yamaguchi *et al.* [87] convert optical flow estimation task into a 1D search problem by using precomputed fundamental matrices and the small motion assumptions. In their methods the dynamic regions are treated as outliers. Instead, Wulff *et al.* [88] use semantic information to split the scene into dynamic objects and static background and only apply strong geometric constraints on the static background.

Recently, end-to-end learning based deep optical flow approaches have shown their superiority in learning optical flow. Given a large amount of training samples, optical flow estimation is formulated to learn the regression between an image pair and corresponding optical flow. These approaches achieve comparable performance to state-of-the-art conventional methods on several benchmarks while being significantly faster. FlowNet [77] is a pioneer in this direction, which needs a large-size synthetic dataset to supervise network learning. FlowNet2 [78] greatly extends FlowNet by stacking multiple encoder-decoder networks one after the other, which could achieve a comparable result to conventional methods on various benchmarks. Recently, PWC-Net [80] combines sophisticated conventional strategies such as pyramid, warping and cost volume into network design and set the state-of-the-art performance on KITTI [89, 90] and MPI Sintel [68]. These deep optical flow methods are hampered by the need for large-scale training data with ground truth optical flow, which also limits their generalization ability.

1.2.5 Depth And Camera Motion From A Monocular Video

The task of recovering the depth and the camera motion from a monocular video (also known as structure from motion) severs as the foundation of 3D reconstruction and visual simultaneous localization and mapping (vSLAM) and has a wide range of applications, including autonomous driving, augmented/virtual reality, indoor navigation, and robotics. Figure 1.5 illustrate the general concept of this task.

The origin of this problem can be traced back to the early 1980s that Longuet-Higgins [91] proves that the camera poses as well as the depth maps can be computed

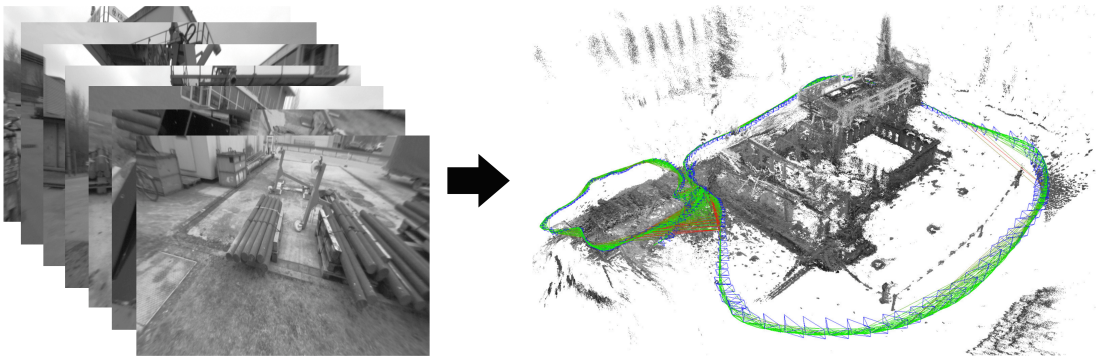


Figure 1.5: **Illustration of structure from motion.** The structure from motion task is trying to recover camera poses and depth from a monocular video.

from image matching points alone without any other information. Mathematically, given a set of image matching points in homogeneous coordinates, $\mathbf{x}_i = \begin{bmatrix} x_i & y_i & 1 \end{bmatrix}^\top$ and $\mathbf{x}'_i = \begin{bmatrix} x'_i & y'_i & 1 \end{bmatrix}^\top$ with known camera intrinsic matrix $\mathbf{K}$, the SfM task is to find a camera rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ as well as the corresponding 3D homogeneous points $\mathbf{X}_i$ and $\mathbf{X}'_i$ such that:

$$\mathbf{x}_i = \mathbf{K} \begin{bmatrix} \mathbf{I} \mid \mathbf{0} \end{bmatrix} \mathbf{X}_i \quad \mathbf{x}'_i = \mathbf{K} \begin{bmatrix} \mathbf{R} \mid \mathbf{t} \end{bmatrix} \mathbf{X}'_i \qquad \forall i. \tag{1.1}$$

A classical method to solve this problem consists of three consecutive steps: 1) Computing the essential matrix $\mathbf{E}$ from the image matching points $\mathbf{x}_i$ and $\mathbf{x}'_i$; 2) Extracting the relative camera pose $\mathbf{R}$ and $\mathbf{t}$ from the essential matrix $\mathbf{E}$; 3) Triangulating the matching points $\mathbf{x}_i$ and $\mathbf{x}'_i$ with the camera pose to get 3D points $\mathbf{X}_i$ and $\mathbf{X}'_i$.

The essential matrix $\mathbf{E}$ can be solved with at least 5 matching points using the equation below:

$$\mathbf{x}'^\top_i \mathbf{K}^{-\top} \mathbf{E} \mathbf{K}^{-1} \mathbf{x}_i = 0 \qquad \forall i. \tag{1.2}$$

$\mathbf{R}$ and $\mathbf{t}$ can be computed from $\mathbf{E}$ using matrix decomposition such that $\mathbf{E} = \mathbf{SR}$, where $\mathbf{S}$ is a skew symmetric matrix and $\mathbf{R}$ is a rotation matrix. Since for any non-zero scaling factor α, $\begin{bmatrix} \alpha \mathbf{t} \end{bmatrix}_\times \mathbf{R} = \alpha \begin{bmatrix} \mathbf{t} \end{bmatrix}_\times \mathbf{R} = \alpha \mathbf{E}$ provides a valid solution, there is a scale ambiguity for relative camera pose estimation. The 3D points $\mathbf{X}_i$ and $\mathbf{X}'_i$ can be computed by triangulation with a global scale ambiguity. A comprehensive analysis and review of this task can be found in [92].

Conventional Methods With decades of development and refinement, the classic standard pipeline [92] is widely used in many conventional state-of-the-art SfM and vSLAM systems [93, 94, 95]. Since all geometry information can be recovered from image matching points, the key is to recover a set of (sparse or dense) accurate matching points. To this end, the pipeline often starts with sparse (or semi-dense) distinct feature extraction and matching to get sparse matching points, as sparse / semi-dense matching is more accurate than dense matching. To further refine the matching results, the RANSAC scheme [96] is used to filter the matching points that do not fit the majority motion. These outliers often include mismatches and dynamic

objects in a scene. After retrieving the camera poses from the refined matching points, the depth of these points can be computed via triangulation. In some cases, if it is desired to estimate dense depth maps rather than the sparse 3D points, multi-view stereo matching algorithms can be used to recover the dense depth maps with the estimated camera poses.

Deep Learning Methods Deep learning technique is introduced to this task in 2017 [97, 98]. Existing deep learning based methods either formulate the problem as pose regression and monocular depth estimation or as pose regression and multi-view stereo matching.

The formal one consists of a monocular depth estimation network and a pose regression network. The geometry constraints are used as self-supervisory signals to regularize both camera poses and depth maps [97, 99, 100, 101, 102]. Because single-view depth estimation is inherently ill-posed, these methods are fundamentally limited by how well they can solve that challenging problem. For example, SfMLearner [97] simultaneously estimates an explainability mask to exclude the dynamic objects while GeoNet [99] utilizes an optical flow module to mask out these outliers by comparing the rigid flow (computed by camera poses and depth maps) with the non-rigid flow (computed by the optical flow module). Other methods focus on implementing more robust loss functions such as ICP loss [100], motion segmentation loss [102], or epipolar loss [101].

The later one requires two image frames to estimate depth maps and camera poses at test time. As a pioneer of this type, DeMoN [98] concatenates a pair of frames and uses multiple stacked encoder-decoder networks to regress camera poses and depth maps, implicitly utilizing multi-view geometry. Similar strategies have been adapted by [103, 104, 105, 106] by replacing generic layers between camera poses and depth maps with optimization layers that explicitly enforce multi-view geometry constraints. For example, BANet [103] parameterizes dense depth maps with a set of depth bases and imposes bundle adjustment as a differentiable layer into the network architecture. Wang *et al.* [105] use regressed camera poses to constrain the search space of optical flow, estimating dense depth maps via triangulation. DeepV2D [106] separates the camera pose and depth estimation, iteratively updating them by minimizing geometric reprojection errors. Similarly, DeepSFM [107] initiates its pose estimation from DeMoN [98], sampling nearby pose hypotheses to bundle adjust both poses and depth estimation.

1.3 Self-supervised Learning

The concept of self-supervised learning can be traced back to 1989 [108] and is applied to computer vision in 1994 [9] for the image classification task. Self-supervised learning is a machine learning technique that tries to train a neural network with automatically generated labels. The intuition behind self-supervised learning is straightforward. Manually labeling a large dataset is laborious and time-consuming, and we can avoid this step by using this technique. There are multiple ways for a

network to learn from unlabeled data, a popular one is to train a network for some pretext tasks first and then apply to downstream tasks. The pretext task is defined as a pre-designed task where its objective function can be used to train a network. There are various pretext tasks such as colourizing grayscale image [10], image inpainting [11], image jigsaw puzzle [12], classifying corrupted images [13], and *etc.* The downstream task is an objective application that can evaluate the performance of the self-supervised network. It is often a high level computer vision task such as object detection, semantic segmentation, image classification, human action recognition, and *etc.* The pretext task aims to learn a good feature representation from unlabeled data and the downstream task reload the features and fine-tune them to a specific task in a supervised/self-supervised way.

Self-supervised learning is slightly different in visual geometry tasks. In high level computer vision tasks, the high level information such as semantics and objects is based on human knowledge, whereas, in visual geometry tasks, the geometry information is *inherent* in images. Therefore, there is no difference between pretext tasks and downstream tasks in self-supervised visual geometry problems. Both of them can be solved with the same objective function without the need of ground truth labels.

The self-supervised learning is introduced to visual geometry tasks in 2016, *e.g.,* monocular depth estimation [109], structure from motion [97], stereo matching [14], and optical flow [110, 111]. For depth estimation tasks, similar to conventional methods, photometric warping errors are proposed to drive the network in an self-supervised way, *e.g.,* [112, 109, 97, 113, 14]. Zhou *et al.* [97] propose an unsupervised method which is iteratively trained via warping error propagating matches. The authors adapted TV (total variation) constraint to select training data and discard uninformative patches. Inspired by recent advances in direct visual odometry (DVO), Wang *et al.* [114] argue that the depth CNN predictor can be learned without a pose CNN predictor. Luo *et al.* [115] reformulate the problem of monocular depth estimation as two sub-problems, namely a view synthesis procedure followed by standard stereo matching.

For optical flow, instead of using ground truth flow as supervision, Yu *et al.* [110] and Ren *et al.* [111] suggest that, the image warping loss can be used as supervision signals in learning optical flow. However, there is a significant performance gap between their work and the conventional ones. Then, Simon *et al.* [116] analyze the problem and introduce a bidirectional census loss to handle illumination variation between frames robustly. Concurrently, Yang *et al.* [117] propose an occlusion-aware warping loss to exclude occluded points in error computation. Very recently, Janai *et al.* [118] extended two-view optical flow to multi-view cases with improved occlusion handling performance.

1.4 Thesis Outline

Based on the problem setting, visual geometry recovery can be divided into four sub-tasks, including depth completion, stereo matching, depth from a single image, optical flow and structure from motion. The thesis is divided into five parts and each part addresses one sub-task as follows.

Part I: Learning depth completion

We propose two traditional depth completion methods. In Chapter 2, we assume that depth maps of a general scene lie on low dimensional sub-spaces. Given a set of sparse depth points, we can regress to a dense depth map using a set of full-resolution principal depth bases. These principal components of natural depth fields are learned from computed depth maps and the dense depth maps can be compactly approximated by a weighted sum of these bases. We propose a unified depth completion algorithm using learned principal component analysis (PCA) bases and can be efficiently solved in a closed form solution. In Chapter 3, we add a piecewise planer model together with the previous PCA bases in depth completion task and solve it through tree-reweighted message passing (TRW-S) [119] scheme.

Part II: Learning depth from a stereo camera

We present a self-supervised stereo matching network which can be trained without the need of ground truth disparities (depths). Similar to conventional methods, we leverage a general illumination consistency assumption that a warped left/right image should be identical to the original corresponding left/right image. A convolutional Long Short-Term Memory (LSTM) module has been proposed to handle temporal information in stereo videos. A new efficient stereo matching architecture is also proposed that can process a pair of 540p stereo images over 100 FPS. To relief human burden in network architecture design, we propose a hierarchical neural architecture search scheme for deep stereo matching and achieve state-of-the-art accuracy on various benchmarks. We also investigate a scenario that a LiDAR and a stereo camera co-exist which is very common in autonomous driving vehicles. We propose a noise aware unsupervised LiDAR-stereo fusion network to automatically correct error LiDAR points and predict accurate and dense depth maps.

Part III: Learning depth from a single image

In this part, we study a problem of recovering depth from a single mixture image without ground truth depth maps. It can be done by decoupling a mixture image into a pair of stereo images and then computing disparities from them. Depending on the mixture type, we examine our method on red-cyan, double vision and monocular images.

Part IV: Learning dense correspondence from a monocular camera

In this part, we propose an unsupervised optical flow network that utilizes global epipolar constraint for both stationary and dynamic scenes. For stationary scenarios, we use the fundamental matrix constraint. For stationary scenarios, we utilize our novel low-rank constraint as well as the union-of-subspace constraint to handle dynamic objects.

Part V: Learning depth and camera motion from a monocular video

In this part, we revisit the problem of deep two-view SfM and propose a new deep SfM pipeline that follows the classic pipeline in which features are matched between image frames to yield *relative* camera poses, from which *relative* depths are estimated. By combining the strengths of deep learning within a classic pipeline, we are able to avoid the ill-posedness in previous deep SfM methods, which allows our approach to achieve state-of-the-art results on several benchmarks in both relative pose estimation and depth estimation.

Part II

Learning depth completion

Chapter 2

Efficient Depth Completion Using Learned Bases

In this chapter, we propose a new global geometry constraint for depth completion. By assuming depth maps often lay on low dimensional subspaces, a dense depth map can be approximated by a weighted sum of full-resolution principal depth bases. The principal components of depth fields can be learned from natural depth maps. The given sparse depth points are served as a data term to constrain the weighting process. When the input depth points are too sparse, the recovered dense depth maps are often over smoothed. To address this issue, we add a colour-guided auto-regression model as another regularization term. It assumes the reconstructed depth maps should share the same nonlocal similarity in the accompanying colour image. Our colour-guided PCA depth completion method has closed-form solutions, thus can be efficiently solved and is significantly more accurate than PCA only method. Extensive experiments on KITTI and Middlebury datasets demonstrate the superior performance of our proposed method.

2.1 Introduction

Acquiring dense and accurate depth measurements is crucial for various applications such as autonomous driving [120], indoor navigation [121], robot SLAM[122], virtual or augmented reality [123]. However, due to the limitation of current depth sensing technology, captured depth maps are often in a sparse form (*i.e.,* LiDAR) or suffering severe data missing problem on transparent and reflective surfaces (*i.e.,* Microsoft Kinect, Intel RealSense, and Google Tango). How to effectively interpolate these sparse depth points becomes an active topic recently.

Depending on the input modalities, the depth completion task can be subdivided to depth inpainting, depth super-resolution and depth reconstruction from sparse samples. Lots of works have been proposed to address this problem. For conventional methods, researchers often employ Markov Random Fields (MRF) to interpolate sparse depth points by encouraging nearby points to have similar depth values. The main drawback is that these methods often require several seconds to minutes to process a single frame and may create lots of artifacts on recovered depth maps.

Deep learning based methods can produce a dense depth map in real time but it often requires lots of training data and suffers generalization issues.

In this chapter, we propose a general geometry constraint for the depth completion task. It is based on an observation that a generic depth fields can be approximated by a set of low dimensional principle component bases that extracted from natural depth maps. We then assume the recovered dense depth map should be a weighted sum of these bases that satisfy the input sparse depth points. Moreover, in order to deal with the over-smooth problem of the recovered depth maps, we further introduce an auto-regression model to allow the accompanied colour images to guide the reconstruction process. Our method has closed form solutions that can be efficiently solve within several seconds. Moreover, since we do not make any assumptions to the modality of the input, our method can be applied to any kind of depth sensors. Extensive experiments on KITTI dataset and Middlebury dataset show that our method is robust to the sparsity of the input and have good cross dataset performance. Moreover, our method can work well even when the input depth points are corrupted.

2.2 Background

Depth completion task can be roughly classified into two categories: conventional methods and deep learning based methods. Here we only review the most related methods.

2.2.1 Conventional methods

Diebel *et al.* [24] presented a Markov Random Fields (MRF) which uses colour information and depth information where available. The underlying assumption is that areas of constant colour are most likely to have constant depths. The work [25] follows the same assumption and compared five different interpolation methods with [24]. More recently [26] proposed a flexible method that is able to up-convert the range sensor data to match the image resolution. However, it is not a per-frame algorithm that requires as least 2 successive frames. In order to intensively use the structural correlation between colour and depth pairs in colour guided depth super-resolution, non-local means (NLM) was introduced as a high-order term in regularization [27]. [28] swapped the Gaussian kernel in the standard NLM to a bilateral kernel to enhance the structural correlation in colour-depth pairs and proposed an adaptive colour-guided auto-regressive (AR) model that formulates the depth upsampling task as AR prediction error minimization, which owns a closed-form solution. A few approaches employ sparse signal representations for guided upsampling making use of the Wavelet domain [29] , learned dictionaries [30] or co-sparse analysis models [31]. [32] and [33] leverage edge-aware image smoothing techniques and formulate it as a weighted least squares problem while [33] also applied coarse-to-fine strategy to deal with the very sparse situation.

2.2.2 Deep learning based methods

Recently, deep learning based end-to-end methods were introduced to image super-resolution [36]. Song *et al.* [37] extend it to the problem of depth map super-resolution. Riegler *et al.* [38] proposed a deeper network for depth map super-resolution that makes the training process much faster with better performance. However, the main drawback for adopting deep learning to our task is the irregular pattern that makes the correlation between the depth points and their spatial locations vary from frame to frame. Very recently, Sparsity Invariant CNNs [35] has been proposed to handle the sparse and irregular inputs. With the guidance of corresponding colour images, Ma *et al.* [124] extended the up-projection blocks proposed by [125] as decoding layers to achieve full depth reconstruction. Jaritz *et al.* [39] presented a method to handle sparse inputs of various densities without any additional mask input. Rather than training an end-to-end neural network to perform depth upsampling, several algorithms leverage deep neural networks to provide high-level cues in improving the performance, i.e., pixel-level semantic cues. With very similar objective as our task, Schneider *et al.* [40] utilizes semantic input provided by fully convolutional networks (FCNs) [126] and achieves relatively good results. However, since their algorithm relies on semantic segmentation, their performance is largely depended on the performance of segmentation results.

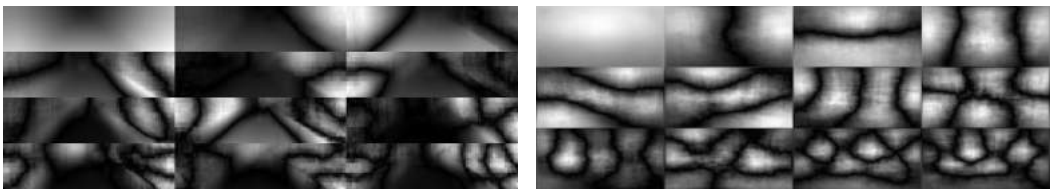
2.3 Problem statement

We define a general depth completion task as follows. Let us define a set of 3D points that measured from a depth sensor as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, where $\mathbf{x}_i = (x_i, y_i, z_i, 1)^T$ is the 3D coordinate of the i^{th} point in homogeneous coordinate. $\mathbf{P}$ is the 3×4 camera projection matrix, which projects 3D points in the world coordinate to 2D image coordinates on the image plane. $\mathbf{T}$ is a 4×4 external matrix that maps 3D points from the sensor's local frame coordinates to world coordinates. The depth measurements $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_n\}$ on the image plane can be derived as $\mathbf{S} = \mathbf{PTX}$, where $\mathbf{s}_i = (u_i, v_i, d_i)^T$, (u_i, v_i) is the i^{th} point's corresponding 2D image coordinate and d_i represents its depth value. The target of this task is to assign a proper depth d_j for each image point (u_j, v_j).

Besides the input sparse depth measurements, we could potentially have other information such as the corresponding high-resolution colour image. With the help of the colour guidance, we expect to predict more accurate depth value d_j for each pixel (u_j, v_j) on the image plane.

2.4 Method

In this section, we propose two depth completion methods. We first formulate the depth maps as a weighted sum of PCA bases and try to solve the problem in a closed form solution. Then we add a colour guided auto-regression model to the formulations to boost the performance.



(a) PCAs from KITTI (b) PCAs from NYU

Figure 2.1: **The extracted first 12 PCA components.**

We use the following priors on depth maps and their accompanied colour images:

1. The depth maps and the colour images have strong local correlations, *i.e.,* the depth map could be expressed/predicted by the colour image in local region;
2. The depth map lies in a low dimensional subspace, *i.e.,* each depth map could be represented as a linear combination of basis depth maps.

2.4.1 Learning Depth Map Bases

We extract PCA bases of natural depth maps with the widely used robust PCA [127] algorithm. Note that it is possible to use other basis learning methods or sparse coding methods to generate the bases. Considering the scale differences between outdoor and indoor scenes, we train two set of PCA bases, one from Sequence 10 of KITTI visual odometry dataset (Figure 2.1 (a)) for outdoor scenarios and the other from NYUV2 training dataset [128] (Figure 2.1 (b)) for indoor scene. Since the ground truth depth maps are incomplete, we inpaint the incomplete depth maps [129] before extracting the PCA bases.

2.4.2 PCA-based Depth Prediction

The PCA based depth prediction method consists of the following steps:

1) Extract the first k basis $\mathbf{A}$ from pre-existing dense depth maps using robust PCA [127];

2) Define the n control points based on depth measurements $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, where $\mathbf{x}_i = (u_i, v_i, d_i)^T$;

3) The problem of dense depth prediction now can be formulated as a least squares fitting problem:

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left\| \widehat{\mathbf{A}}\mathbf{w} - \mathbf{X} \right\|_2^2 \tag{2.1}$$

where $\mathbf{w} = (\mathbf{w_1}, \mathbf{w_2}, ...\mathbf{w_k})^T$ defines the weights and $\widehat{\mathbf{A}}$ is a subset of the PCA bases $\mathbf{A}$ with $n \times k$ dimensions, corresponding to the positions of available measurements. There exists a closed-form solution for the above least squares problem

as:

$$\mathbf{w} = \widehat{\mathbf{A}}^{\dagger}\mathbf{X}, \tag{2.2}$$

where $\widehat{\mathbf{A}}^{\dagger}$ denotes the pseudo-inverse of $\widehat{\mathbf{A}}$.

4) Dense depth $\mathbf{d}$ can be predicted as:

$$\mathbf{d} = \mathbf{A}\mathbf{w}. \tag{2.3}$$

Note that we do not use weighted PCA as we assume all sparse depth measurements are with the same confidence.

2.4.3 Colour-guided PCA for Depth Prediction

Colour-guided smoothness term, aims at representing the local structure of the depth map. Depth maps for generic 3D scenes contain mainly smooth regions separated by curves with sharp boundaries. The key insight behind the colour guided smoothness term is that the depth map and the colour image are locally correlated, thus the local structure of the depth map can be well represented with the guidance of the corresponding high-resolution colour image. The term is widely used in image colourization, depth in-painting and depth image super resolution. We add this term together with the previously proposed global geometry constraint term to further boost the performance.

Denote D_u as the depth value at location u in a depth map, the depth map inferred by the model can be expressed as:

$$D_u = \sum_{v \in \theta_u} \alpha_{(u,v)} D_v, \tag{2.4}$$

where θ_u is the neighbourhood of pixel u and $\alpha_{(u,v)}$ denotes the colour guided smoothness model coefficient for pixel v in the set of θ_u. The discrepancy between the model and the depth map (the colour guided smoothness potential) can be expressed as:

$$\psi_\theta(\mathbf{D}_\theta) = \left(D_u - \sum_{r \in \theta_u} \alpha_{(u,v)} D_v \right)^2. \tag{2.5}$$

We need to design a local colour guided smoothness predictor α with the available colour image.

$$\alpha_{(u,v)} = \frac{1}{N_u} \alpha^I_{(u,v)} \tag{2.6}$$

where N_u is the normalization factor, $\overline{D}$ is the observed depth map, I is the corresponding colour image.

$$\alpha^I_{(u,v)} = \exp\left(-\sum_{i \in C} \|\mathbf{B_u} \circ (g_u - g_v)\|_2^2 / (2 \times 3 \times \sigma^2)\right) \tag{2.7}$$

where g represents the intensity value of corresponding colour pixels, σ is the vari-

ance of colour intensities in the local path around u. "$\circ$" denote the element-wise multiplication. $\mathbf{B_u}$ represents the bilateral filter kernel: $\mathbf{B_u}_{(u,v)} = \exp(-\sum_{i\in C}(g_u - g_v)^2/(2\times 3\times \sigma_{I_u}^2))$, where σ_{I_u} controls the importance of intensity difference. The window size for θ is set as 9×9 in our experiment.

With the help of colour-guided auto-regression model [28], we can formulate the problem as a convex minimization with respect to $\mathbf{d}$ and $\mathbf{w}$ simultaneously,

$$\min_{\mathbf{d},\mathbf{w}} \frac{1}{2}\left\|\mathbf{P}\mathbf{d}-\mathbf{d^0}\right\|_2^2 + \frac{\lambda}{2}\left\|\mathbf{Q}\mathbf{d}-\mathbf{d}\right\|_2^2 + \frac{\gamma}{2}\left\|\mathbf{A}\mathbf{w}-\mathbf{d}\right\|_2^2 \tag{2.8}$$

where $\mathbf{P}$ defines the observation matrix, and $\mathbf{Q}$ is a prediction matrix corresponding to colour guided smoothness predictors $\alpha_{(u,v)}$. The above optimization problem also owns a closed-form solution.

Let's denote $\mathbf{P'} = [\mathbf{P},\mathbf{0}], (\mathbf{I}-\mathbf{Q})' = [\mathbf{I}-\mathbf{Q},\mathbf{0}], \mathbf{A'} = [\mathbf{I},-\mathbf{A}], \mathbf{x} = [\mathbf{d},\mathbf{w}]$, then the above equation (2.8) can be expressed as:

$$\arg\min_{\mathbf{x}} \frac{1}{2}\left\|\mathbf{P'}\mathbf{x}-\mathbf{d^0}\right\|_2^2 + \frac{\lambda}{2}\left\|(\mathbf{I}-\mathbf{Q})'\mathbf{x}\right\|_2^2 + \frac{\gamma}{2}\left\|\mathbf{A'}\mathbf{x}\right\|_2^2. \tag{2.9}$$

The closed-form solution to the above optimization is achieved as:

$$(\mathbf{P'}^T\mathbf{P'} + \lambda(\mathbf{I}-\mathbf{Q})'^T(\mathbf{I}-\mathbf{Q})' + \gamma\mathbf{A'}^T\mathbf{A'})\mathbf{x} = \mathbf{P'}^T\mathbf{d}^0 \tag{2.10}$$

2.5 Evaluation

We evaluate our method extensively on well known KITTI stereo and visual odometry (VO) datasets and Middlebury datasets with a comparison to current state-of-the-art approaches.

2.5.1 Error Metrics

We deploy two quantitative measurements:

1) **Mean relative error** (MRE), which is defined as $\mathrm{e}_{rel} = \frac{1}{N}\sum_{i=1}^{N}\frac{|d_i-\widehat{d}_i|}{d_i}$, where d_i and $\widehat{d}_i$ are the ground truth depth and depth prediction respectively. A lower MRE indicates a better dense depth prediction performance achieved.

2) **Bad pixel ratio** (BPR) measures the percentage of erroneous positions in total, where a depth prediction result is determined as erroneous if the absolute depth prediction error is beyond a given threshold d_{th}. A lower BPR indicates a better depth prediction results.

BPR and MRE measure different statistics of the dense depth prediction results, which jointly evaluate the prediction performance.

(a) The input: downsampled Lidar points

(b) Ground truth depth map provided by the 64-line Velodyne HDL-64E LIDAR.

Figure 2.2: **Experiment setting:** Taking sparse and noisy depth measurement in (a) as input, with the learned depth map bases and the guidance of colour image, our method outputs a dense depth map, which is evaluated against the ground truth depth map illustrated in (b).

Table 2.1: **Evaluation on the KITTI stereo 2012 dataset**

Percentage	PCA	AR [28]	Colour Guide PCA
Relative Error	5.66	10.43	**3.33**
Bad Pixel Ratio	18.59	13.15	**10.19**

2.5.2 Results on the KITTI dataset.

We perform evaluation of two subset of KITTI dataset: KITTI stereo and KITTI VO, which both consist of challenging and varied road scene imagery collected from a test vehicle. Ground truth depth maps are obtained from 64-line LIDAR data. The main difference between these two is that the ground truth depth maps from the stereo dataset were manually corrected and interpolated based on their neighborhood frames. Note we only used the lower half part of the images (200×1226) as the upper half part generally include large part of sky and there is no depth measurements available. We down-sampled ground truth depth map with stride of 8 as the input. Our algorithm will recover a dense depth map but we only evaluate its performance on the original sparse depth points. An example of input and ground truth depth map is given in Figure 2.2.

In Table 2.1 and Table 2.2, we compare our method (colour Guide PCA) to state-of-the-art colour guided depth interpolation approach [28] on the KITTI 2012 and 2015 datasets, respectively. In Bad Pixel Ratio, we use threshold of 3 meters. Our method outperforms current state-of-the-art results on both dataset with a notable margin. Our method achieves 7.10 and 8.63 performance leap for KITTI 2012 and 2015 datasets respectively.

We also perform the quantitative and qualitative evaluation of our method on KITTI VO dataset. KITTI VO dataset consists of 22 sequences 43,596 frames which

Table 2.2: **Evaluation on the KITTI stereo 2015 dataset**

Percentage	PCA	AR [28]	Colour Guide PCA
Relative Error	7.50	13.53	**4.90**
Bad Pixel Ratio	18.48	14.71	**11.84**

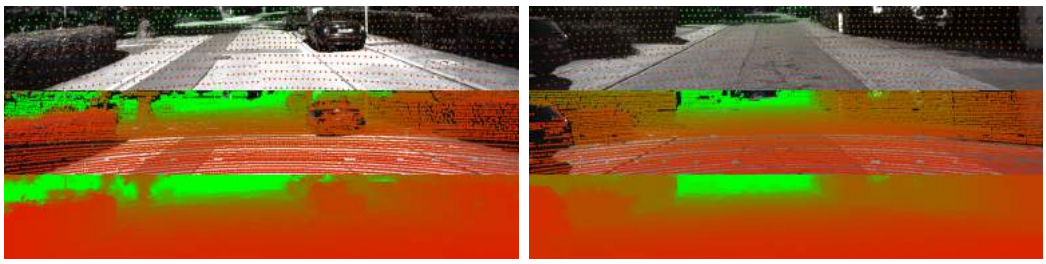
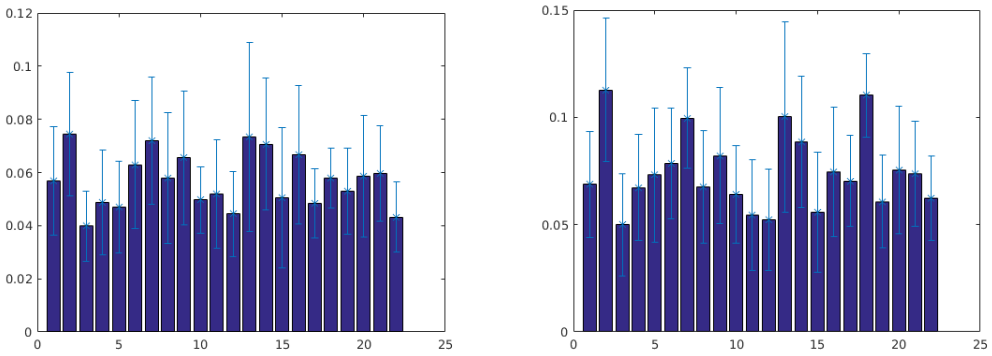


Figure 2.3: **Recovery results on KITTI VO dataset.** top: the input sparse Lidar points; middle: the ground truth Lidar points; bottom: the recovered dense depth map.



(a) Mean relative error on each sequence.

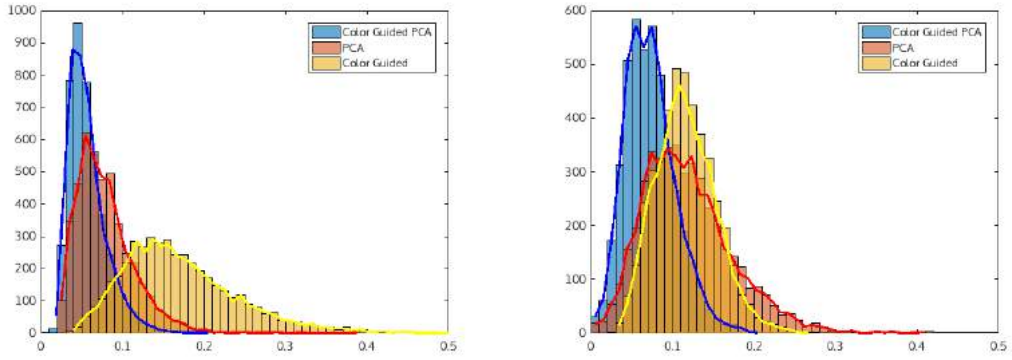
(b) Bad pixel ratio on each sequence with threshold 3 meters

Figure 2.4: **Quantitative results on each sequences.** The error bar shows the standard deviation.

including vary driving scenarios such as highway, city and country road. In our experiment setting, we only use the left images and Lidar points. Note, we recover the depth every 10 frames, 4,359 frames in total.

Figure 2.3 illustrates two sample results. Note that our method even successfully recovers the correct depth on car windows which lacks the depth measurement in ground truth depth map.

The quantitative results are shown by sequence in Figure 2.4 and by histogram in Figure 2.5. The overall Mean relative error is 0.0569 and the Bad pixel ratio is 0.0725 with a threshold of 3 meters. Figure 2.5 also provides the comparison between PCA and colour guided method. Our method performs notable better through out the whole KITTI VO dataset.



(a) Histogram of Mean relative error on KITTI VO dataset

(b) Histogram of Bad pixel ratio on KITTI VO dataset

Figure 2.5: **Quantitative comparison with PCA and colour guidance methods.** Our method notably outperforms all of them.

2.5.3 Results on the Middlebury dataset.

Our proposed method can also be used as a standard depth upsampling method. We conduct evaluation on the Middlebury dataset with 8× upsampling rate. The resolution of colour guide and ground truth disparity map is set to 640×480. In quantitative comparison, as there are holes in ground truth disparity map, we exclude these holes areas in comparison. Note we use threshold of 1 pixel in Bad Pixel Ratio metric.

Quantitative comparison with state-of-the-art method [130] is shown in Table 2.3 and 2.4 with Mean relative error and Bad pixel ratio respectively. Our method significantly outperform the state-of-the-art in all aspects with average margin 31.21% in MRE and 69.67% in BPR.

Figure 2.6 illustrates two sample results. Our method produce much sharper and more accurate boundaries. The texture on background board and the words in front board do not misguide our algorithm to generate texture copy effects on the recovered depth map.

Table 2.3: **Evaluation in MRE on the Middlebury dataset**

Percentage	Art	Books	Dolls	Laundry	Moebius	Reindeer
Bicubic	3.69	1.42	1.69	2.20	1.82	2.13
Bilinear	3.63	1.28	1.53	2.00	1.64	1.96
Ferstl[130]	2.76	1.07	1.28	1.93	1.43	1.56
Our method	**2.25**	**0.98**	**0.92**	**1.35**	**1.03**	**1.16**

Table 2.4: **Evaluation in Bad pixel ratio on the Middlebury dataset**

Percentage	Art	Books	Dolls	Laundry	Moebius	Reindeer
Bicubic	25.42	9.93	13.15	15.73	12.73	13.39
Bilinear	20.90	8.41	12.44	13.98	11.82	11.01
Ferstl[130]	16.98	11.19	14.31	15.76	12.66	11.51
Our method	**12.75**	**7.49**	**7.83**	**7.33**	**7.73**	**6.62**

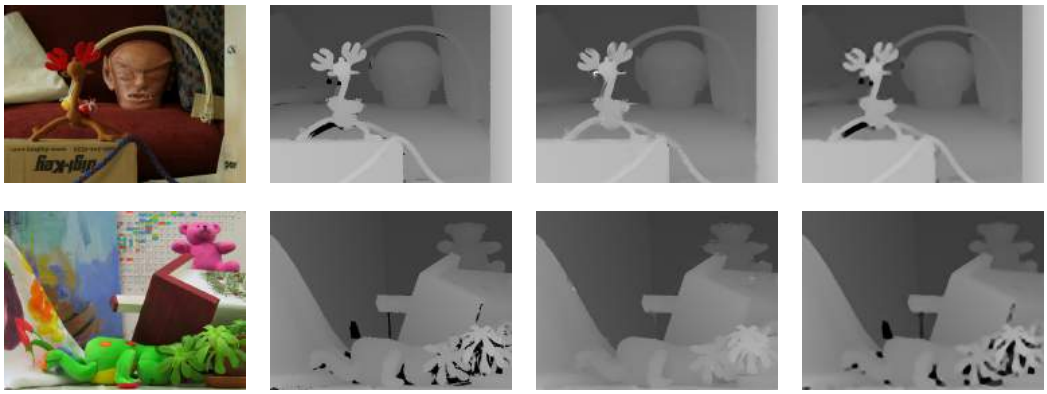


Figure 2.6: **Recovery results on Middlebury dataset.** From left to right: colour images, ground truth depth maps, our results and results from [37].

2.6 Conclusion

In this chapter, we have proposed a novel PCA-based colour guide depth completion algorithm which is able to recover high quality high resolution depth map from sparse inputs in a closed form solution. Experiments on "KITTI" and "Middlebury" dataset demonstrate that our algorithm has high resistance on texture copy effects and significantly outperform state-of-the-art methods with sharper boundaries, including deep learning based methods.

Chapter 3

Depth Completion using Piecewise Planar Model

In the last chapter, we have shown that depth maps can be assembled by a set of learned bases and can be efficiently solved in a closed form solution. However, one issue with this method is that it may create artifacts when colour boundaries are inconsistent with depth boundaries. In fact, this is very common in a natural image. To address this issue, we enforce a more strict model in depth recovery: a piece-wise planar model. More specifically, we represent the desired depth map as a collection of 3D planar and the reconstruction problem is formulated as the optimization of planar parameters. Such a problem can be formulated as a continuous CRF optimization problem and can be solved through particle based method (MP-PBP) [131]. Extensive experimental evaluations on the KITTI visual odometry dataset show that our proposed methods own high resistance to false object boundaries and can generate useful and visually pleasant 3D point clouds.

3.1 Introduction

Autonomous driving requires the vehicles to efficiently sense and understand the surrounding 3D world in real time. Currently, most of the autonomous vehicles (Google, Uber, Ford, Baidu, etc) are equipped with high-end 3D scanning systems such as Velodyne which could provide accurate 3D measurements that are critical to the autonomous vehicles' decision making and planning. However these high-end 3D scanning systems such as Velodyne LIDAR sensors are quite expensive with the cost comparable to the cost of the whole vehicle, which may hinder their admittance to the global consumer market.

To effectively reduce the cost in surrounding 3D world sensing, a natural and cost efficient alternative would be using passive sensors such as monocular cameras or stereo cameras, which could not only provide 3D measurements (by using structure-from-motion (SfM) or simultaneous localization and mapping (SLAM)) but also semantic information (not available from the point clouds). However, the computer vision based system is either not robust (different weather conditions could result in dramatically different vision measurements) or inaccurate (compared with

LiDAR based 3D scanning).

In this chapter, we propose to investigate effective fusion to integrate sparse 3D point clouds (low cost) and high-resolution colour images, such that generating dense 3D point clouds comparable to high-end 3D scanning systems. Specifically, the objective of this chapter is to "**generate/predict a dense depth map and the corresponding 3D point clouds from very sparse depth measurements with/without colour images and colour-depth image datasets**". Under our framework, we use a low-resolution LIDAR and a high-resolution colour image to generate dense depth maps/3D point clouds (Fig. 3.1). This can be understood as using the high-resolution colour image to augment the sparse LIDAR map and predict a dense depth map/3D point clouds. Our proposed methods can not only provide accurate dense depth maps but also provide visually pleasant 3D point clouds, which are critical for autonomous driving in urban scenes. Specifically, we have presented a novel way in perceiving 3D surrounding environments, which owns low-cost compared with the high-end LIDAR sensors and high precision and efficiency compared with the colour camera only solutions. The proposed framework owns great potentials in developing compact and high-resolution LIDAR sensors, at very low cost, for domain-specific applications (e.g. ADAS, autonomous driving).

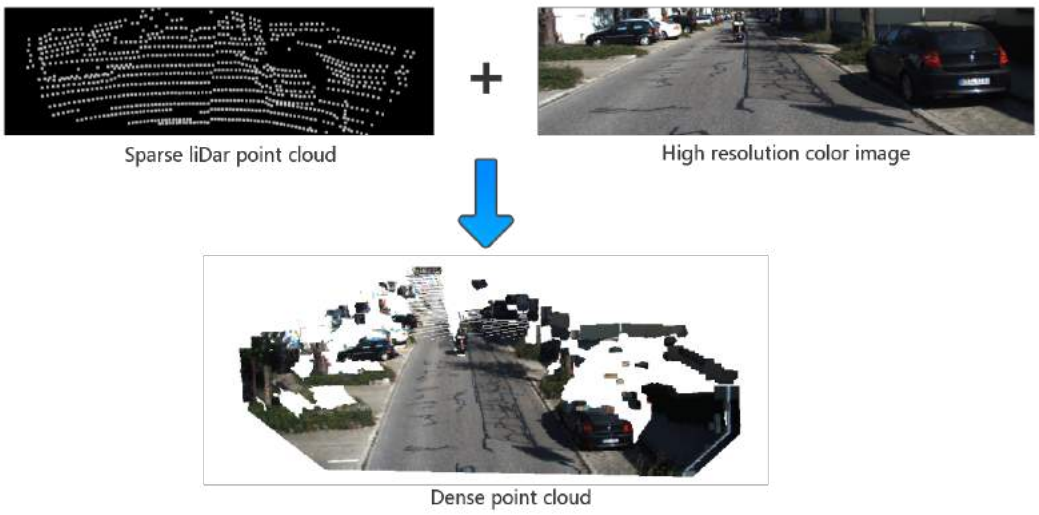


Figure 3.1: Conceptual illustration of the problem of predicting dense 3D point clouds/depth maps from sparse LiDar input, where the input is a collection of sparse 3D point clouds and desired output is a dense 3D point cloud/depth map.

Our dense depth map prediction framework differs from general depth interpolation methods and depth super-resolution methods in the following aspects: 1) Irregular depth points pattern; 2) Very sparse depth measurements; and 3) Missing real obstacles or generating false obstacles in a certain range is unacceptable. The irregular depth points pattern creates a barrier in applying deep convolutional neural network (CNN) based methods as it requests irregular convolution pattern,

which is still an unsolved problem in deep learning. On the other hand, even though many traditional methods [30, 130, 28, 32, 33] could deal with irregular patterns, they suffer from the dependency on the strong correlation between colour images and depth maps, where boundaries that only exist in the colour images will mislead the methods and generate false obstacles on road. For those general smooth interpolation methods such as nearest neighbor, bilinear interpolation, bicubic interpolation and [129], since they neglect the colour images in problem formation, they tend to generate over-smoothed results and miss real obstacles.

In this chapter, targeting at handling the above difficulties with existing methods, we propose a dense depth prediction approach that only uses the boundaries in colour image (i.e., the superpixel over-segmentation). We resort to the piecewise planar model for general scene reconstruction, where the desired depth map/point clouds are represented as a collection of 3D planar and the reconstruction problem is formulated as the optimization of planar parameters. The resultant optimization involves the unary term evaluated at the sparse depth measurements, the smoothness term across neighboring planes. Thirdly, as the urban driving scenarios are well structured, we propose a specifically designed model for urban driving scenario called "cardboard world" model, i.e., front-parallel orthogonal piecewise planar model, where each segment can only be assigned to either the road plane or a front-parallel object plane orthogonal to the road plane. We formulate the problem as a continuous CRF optimization problem and solve it through particle based method (MP-PBP) [131]. Extensive experiments on the KITTI visual odometry dataset show that the proposed methods owns high resistance to false object boundaries and can generate useful 3D point clouds without missing obstacles.

3.2 Approach

Real world driving scenarios generally consist of road, surrounding buildings, vehicles, pedestrians and etc., which can be well approximated with piece-wise planar model in 3D representation. By representing the traffic scenes with piece-wise planar model, we are able to exploit the structural information in the scenes and the number of variables can be greatly reduced (For each plane, we only need 3 parameters to represent.). In this way, the number of measurements of sparse depths could be greatly reduced, which enables us the ability to work with very sparse LIDAR measurements. Furthermore, the parametric model owns the ability to handle outlying LIDAR measurements.

3.2.1 An overview of the proposed method

A high level description of our method is given as follows: assume the LIDAR sensor and the camera have been geometrically calibrated (both intrinsically and extrinsically). Given the input of a single frame monocular image with corresponding sparse depth points, we first perform image over-segmentation to obtain fixed number of super-pixels using the SLIC algorithm [132]. Then we conduct interpolation

on the sparse depth points to generate an initial dense depth map by using the penalized least squares method [129]. This dense depth map is used to provide initial planar parameters for each segment. As shown in Fig. 3.2, on average each superpixel region contains a single depth point measurement. Also note some superpixels do not contain any depth measurement. After fitting the initial depth measurements inside each superpixel with a plane, we formulate a Conditional Random Field (CRF) to optimize all the plane parameters and recalculate the depth map. A flowchart of our approach is given in Fig. 3.3.

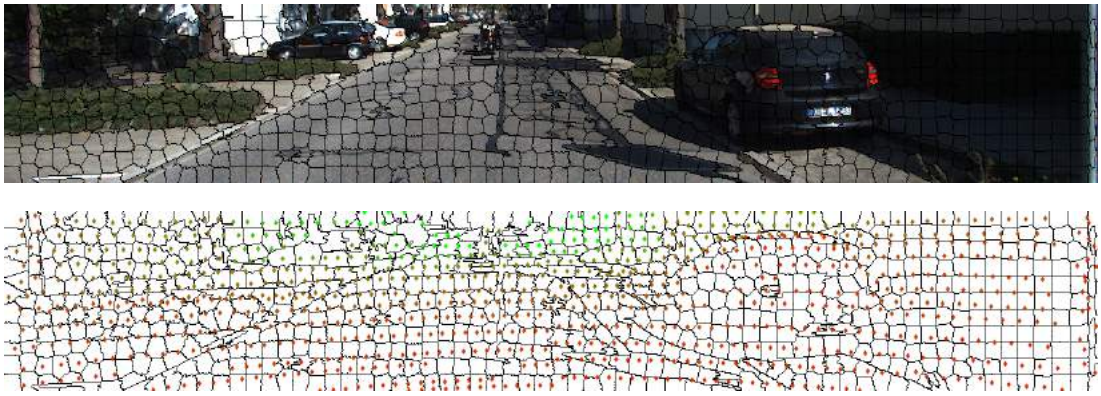


Figure 3.2: **Inputs of our algorithm implementation.** Top: The SLIC oversegmentation of the input image. Bottom: the input of our algorithm: the super-pixel segmentation and sparse depth measurements. Note that as the depth measurements are very sparse, there are only a few depth measurements in each super-pixel and there are considerable super-pixels where no depth measurements are available.

3.2.2 Mathematical Formulation

Under our piece-wise planar model for the scenes, each segment is well approximated with a plane. In this way, the dense depth prediction problem is transformed to the optimization of the plane parameters. Furthermore, we assume that the boundaries in the depth maps are a subset of the boundaries in the colour images, which enables us the freedom to decide the real depth boundaries from the colour boundaries.

Specifically, let $\mathcal{S}$ denote the set of superpixels and each superpixel $i \in \mathcal{S}$ is associated with a segment $\mathcal{R}_i$ in the image plane and a random variable $\mathbf{s}_i = \mathbf{n}_i$, where $\mathbf{n}_i \in \mathbb{R}^3$ describes the plane parameter in 3D. Our goal is to infer the 3D geometry of each superpixel $\mathbf{s}_i$ given the sparse depth measurements $d_i(\mathbf{x})$. We define the energy of the system to be the sum of a data term φ_i and a smoothness term $\psi_{i,j}$,

$$E(\mathbf{s}) = \sum_{i \in \mathcal{S}} \varphi_i(\mathbf{s}_i) + \sum_{i \sim j} \psi_{i,j}(\mathbf{s}_i, \mathbf{s}_j), \tag{3.1}$$

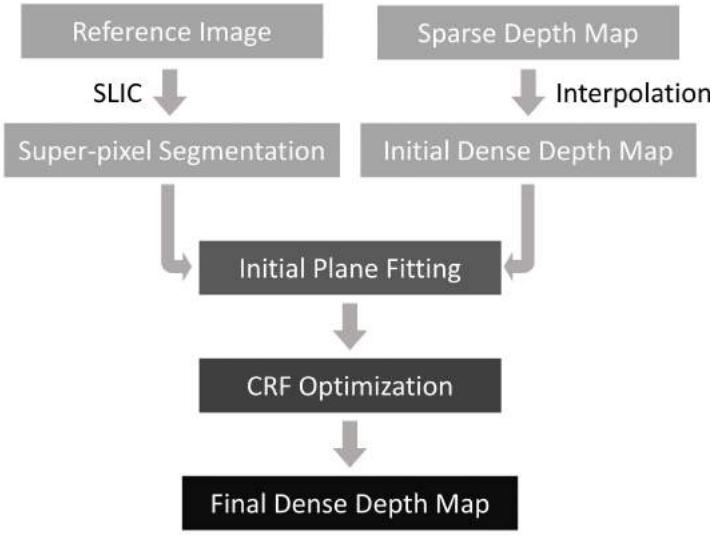


Figure 3.3: **Pipeline of our method.** Given high resolution colour image and sparse depth measurements as input, our approach predicts dense depth map with the same resolution as the colour image.

where φ denotes the data term while ψ denotes the smoothness term. $\mathbf{s} = \{\mathbf{s}_i | i \in \mathcal{S}\}$ and $i \sim j$ denotes the set of adjacent superpixels in $\mathcal{S}$.

Data Term: The data term encourages the depth measurement to lie on the plane by penalizing the discrepancy between the measurement and the prediction. To better enforce this constraint, we choose the ℓ_2 norm to amplify the cost. Therefore, our data term can be written as:

$$\varphi_i(\mathbf{s}_i) = \theta_1 \sum_{\mathbf{x} \in \mathcal{X}} \|\widehat{\mathrm{d}}(\mathbf{s}_i, \mathbf{x}) - \mathrm{d}_i(\mathbf{x})\|_2^2, \tag{3.2}$$

where $\mathcal{X}$ is the set of pixels that has depth measurements. $\widehat{\mathrm{d}}(\mathbf{s}_i, \mathbf{x}) = \frac{-\widetilde{\mathrm{d}}_i}{\mathbf{n}_i^T \mathbf{K}^{-1} \mathbf{x}}$ represents the estimated depth value on a pixel x, $\widetilde{\mathrm{d}}_i$ represents the distance between the plane and the origin, $\mathbf{n}_i^T$ is the normal vector of the plane, $\mathbf{K}$ is the camera intrinsic matrix and $\mathbf{x} = (u, v, 1)^T$ is the homogeneous representation of the pixel x. $\mathrm{d}_i(x)$ is the depth measurement on pixel x.

Smoothness Term: The smoothness term encourages coherence of adjacent superpixels in terms of both depth and orientation. The depth coherence is defined by the difference between the depths of neighboring superpixels' boundaries while the orientation coherence is defined as the difference between the surface normal of neighbouring superpixels. Considering the discontinuity in neighboring superpixels due to scene structure, we use the truncated ℓ_1 norm to allow discontinuity in both depth and orientation.

Algorithm 1 Solving Eq. (3.1) via the PCBP

Input:
Superpixels $\mathbf{S}$ and sparse depth measurements $\mathbf{d}$, number of particles n_p, number of iterations n_i, parameters θ_1, θ_2, θ_3, τ_1, τ_2, ρ, σ.

Initialize: Initial plane parameters $\mathbf{s}_i$ for each superpixel segment.

while iteration $< n_i$ **do**
1). Sample particles: the first particle for each superpixel is the result of previous iteration, the next $n_p/2$ particles are randomly sampled around the state in the previous iteration and the remaining $n_p/2 - 1$ particles are sampled from the neighboring superpixels' current states;
2). Evaluate the data term (3.2) and the smoothness term (3.3);
3). Solve the resultant discrete problem with TRW-S and update plane parameters for each superpixel.
end while

Output: Plane parameters $\mathbf{S}$, recovered depth map $\mathbf{D}$.

Following [90], our smoothness potential $\psi_{i,j}(\mathbf{s}_i, \mathbf{s}_j)$ can be decomposes as:

$$\psi_{i,j}(\mathbf{s}_i, \mathbf{s}_j) = \theta_2 \psi_{i,j}^{depth}(\mathbf{n}_i, \mathbf{n}_j) + \theta_3 \psi_{i,j}^{orient}(\mathbf{n}_i, \mathbf{n}_j), \tag{3.3}$$

with weights θ and

$$\psi_{i,j}^{depth}(\mathbf{n}_i, \mathbf{n}_j) = \sum_{\mathbf{p} \in \mathcal{B}_{i,j}} \rho_{\tau_1}(d(\mathbf{n}_i, \mathbf{p}) - \mathrm{d}(\mathbf{n}_j, \mathbf{p})),$$

$$\psi_{i,j}^{orient}(\mathbf{n}_i, \mathbf{n}_j) = \rho_{\tau_2}(1 - |\mathbf{n}_i^T \mathbf{n}_j| / (|\mathbf{n}_i||\mathbf{n}_j|)),$$

where $\mathcal{B}_{i,j}$ is the set of shared boundary pixels between superpixel i and superpixel j, and ρ is the robust ℓ_1 penalty function $\rho_\tau(x) = \min(|x|, \tau)$.

3.2.3 CRF Optimization

The optimization of the above continuous CRF defined in Eq. (3.1) is generally NP-hard. In order to efficiently solve this optimization problem, we discretize the continuous variables and leverage particle convex belief propagation (PCBP) [133], an algorithm that is guaranteed to converge and gradually approach the optimum. It works in the following way: after initialization, for each random variable, particles are sampled around current states. Then these particles act as labels in discretized MRF/CRF that can be solved by any MRF/CRF solving methods such as multi-label graph cut, sequential tree-reweighted message passing (TRW-S) [119] and update the MAP estimation to current solution. The process is repeated for a fixed number of iterations or until convergence.

Implementation Details: A 3D plane is defined by:

$$aX_i + bY_i + cZ_i + d = 0 \tag{3.4}$$

where (a, b, c, d) are the plane parameters, (X_i, Y_i, Z_i) is the 3D points coordinates that can be computed by

$$X_i = (u_i - C_x) \times Z_i / f, \tag{3.5}$$

$$Y_i = (v_i - C_y) \times Z_i / f, \tag{3.6}$$

where $Z_i = d_i$. (u_i, v_i) is the point coordinate in the image plane, (C_x, C_y) is the camera principal point offset and f is the camera focal length.

Thus, the i^{th} particle is defined as a 4×1 vector $(a_i, b_i, c_i, d_i)^T$. a_i, b_i, c_i, d_i are independently generated through a normal distribution with standard deviation σ and mean μ, where σ is given by user setting and μ is the state in the previous iteration.

Our approach to solve Eq. (3.1) is outlined in Algorithm 1, where parameters θ_1 θ_2 θ_3 τ_1 τ_2 have been already defined in previous equations, ρ is the decay rate for generating particles and σ is a 4×1 vector that contains the standard deviation for the 4 parameters of each particle.

Particle Generation: Instead using of the regular PCBP particle generation scheme, which generates particles only from the MCMC framework, we partially adopt the PMBP [134] scheme by adding neighboring plane parameters into the particles, thus the candidate particles are a mixture of MCMC sampling around the previous states and the states from neighboring planes. The advantage of this modification is that it allows neighboring superpixels to fuse together thus decreasing the energy. For example, the road area is often segmented into several segments. By using the regular PCBP particle sampling strategy, each segment's parameters are independently generated by a normal distribution. Even though the smoothness term encourages the planes be close to each other, there will still exist small gap between them as the algorithm could not find exactly the same parameters from its candidate particles. However, in our modified version, the neighboring superpixels can share their parameters and therefore resolve the issue.

In each PCBP iteration, each superpixel has fixed number of particles, and we need to find the sub-optimal combination that has the lowest energy. This problem can be efficiently solved through tree-reweighted max-product message passing (TRW-S) algorithm. The processing time depends on the number of superpixels and the number of particles.

3.2.4 Cardboard World Model: A More Constrained Model

In the above section, we described our piece-wise planar model for solving the dense depth prediction problem, where each plane has three freedoms in 3D space.

Here we notice that for real-world driving applications, man-made road scenes often have stronger structured information (i.e. stronger prior). In particular, we

realize that modeling a front-view road scene as a combination of ground-plane and many front-parallel obstacles planes will be convenient for the task of drivable free-space detection task. Next, we will show how to infer such a simplified 3D road scene model by using the same method of our CRF–conditional random field framework.

Based on these observations, we propose our "cardboard world" model for representing the driving scenes. Under the "cardboard world" model, there only consist two kinds of planes: the road plane and the object plane. We assume there is only one road plane in the scene and all the other planes are object plane. These two kinds of planes are orthogonal to each other and object planes are front parallel. Fig. 3.4 illustrates an example of our "cardboard world" model.

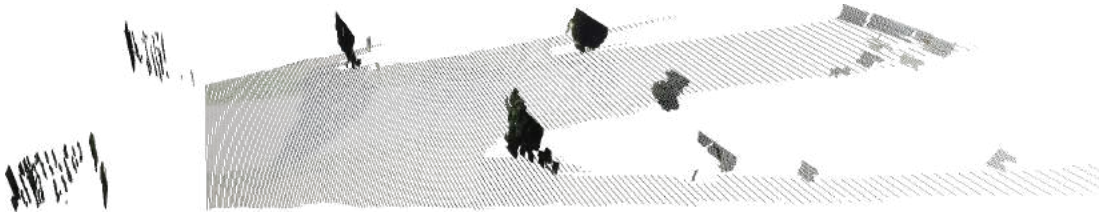


Figure 3.4: **An example of our "cardboard world" model:** Object planes are front parallel planes that orthogonal to the road plane.

There are three advantages in replacing the slanted plane model in the piecewise planar model with our proposed "cardboard world" model:

1) The recovered 3D point clouds are more visually pleasant and the location of each object is more accurate. In some case, when there are two depth points with very different depth values in a single superpixel, the slanted plane model will use a very slanted plane to fit these two points. As a result, the shape of this area will be largely distorted. On the other hand, in the "cardboard world" model, it will force the plane to become front parallel therefore maintain the object shape.
2) As a byproduct, this method provides a free-space for autonomous driving vehicles. We embed a binary labeling problem in our task that classifies superpixels into two clusters: road and object. Superpixels labeled as road belong to the free-space.
3) Processing time can be greatly reduced. By applying a front-parallel constraint and orthogonal constraint, the number of parameters to optimize have been greatly reduced thus decreasing the processing time.

A diagram of our cardboard model approach is shown in Fig. 3.5.

Initialization: Different from the piecewise planar based method, this new method requires an initial road plane estimation for initialization. To do so, we fit the road plane directly from sparse depth point using RANSAC. After getting the initial dense depth map using [129], for each superpixel segment, we project all the depth points into 3D and calculate the distance between them and the road plane by summing the Euclidean distance of each point. If the distance is smaller than a given threshold ϵ, we label the superpixel as road plane and fit it with the initial road plane parameters.

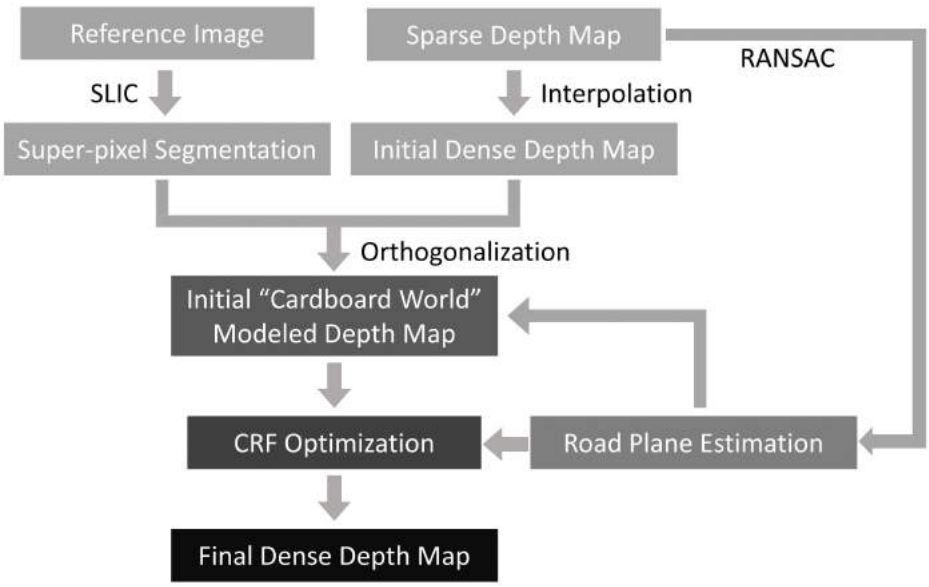


Figure 3.5: **Process flow of our approach.** Given high resolution colour image and sparse depth measurements as input, our approach predicts dense depth map with the same resolution as the colour image.

Otherwise, the superpixel is labeled as object and will be fitted with a front parallel plane with a depth of the mean depth of this superpixel.

The optimization problem remains the same as the previous one. However, since we need to enforce the orthogonal relationship as well as the front parallel constraint, given road plane parameter $(a_r, b_r, c_r, d_r)^T$, the normal of object planes are fixed as well and are equal to $(0, -\frac{c_r}{b_r}, 1)^T$. Hence there is only one freedom in object planes, i.e., the unknown depth. Note that we naturally embed a binary labeling problem in optimization step since superpixels are divided into two groups: road plane group and object planes group. This labeling problem is jointly solved through PCBP process by adding road plane parameters into particle sets of every superpixels, so that every superpixel has a choice to joint the road plane when the fitting cost is low enough. Our approach to solving Eq. (3.1) using our "cardboard world" model is outlined in Algorithm 2.

3.3 Experiments

3.3.1 Experimental setup

We perform both quantitative and qualitative evaluations of our methods on the KITTI VO dataset. The KITTI VO dataset consists of 22 sequences 43,596 frames which includes various driving scenarios such as highway, city and country road. It provides stereo images and semi-dense 360° LIDAR points. In our experimental setting, we only use the left images and the sparse LIDAR points. Note, we recover

Algorithm 2 Solving Eq. (3.1) under "cardboard world" model via the PCBP

Input:
Superpixels **S** and sparse depth measurements **d**, number of particles n_p, number of iteration n_i, parameters θ_1, θ_2, θ_3, τ_1, τ_2, ρ, σ, ϵ.
Initialize: Estimate road plane parameter $\mathbf{s}_d$, initial dense depth map D_0
for all $\mathbf{S_i} \in \mathbf{S}$ **do**
Project every depth points into 3D;
Calculate the sum of Euclidean distance between each 3D point the road plane $\mathbf{s}_d$;
if The sum of Euclidean distance is less than a given threshold ϵ **then**
Fit $\mathbf{S}_i$ with $\mathbf{n}_\mathrm{d}$
else
Fit $\mathbf{S}_i$ with front parallel plane and orthogonal to $\mathbf{n}_\mathrm{d}$
end if
end for
while iteration $< n_i$ **do**
1). Sample particles: the first particle for each superpixel is the state in previous iteration, then randomly generate $n_p/2-1$ particles and add $n_p/2-1$ neighboring superpixels' parameters to the particle set and add the last particle with $\mathbf{n_d}$;

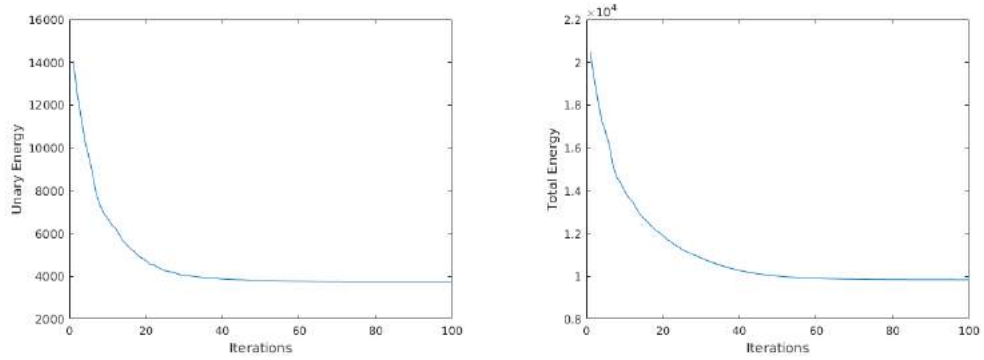
2). Calculate data energy through (3.2) and smoothness energy through (3.3);
3). Solve through TRWS and update current MAP estimation **n** to the first particle;
end while
Output: Plane parameters **n**, recovered depth map **D**.

the depth for the every 10th frames, thus 4,359 frames are recovered in total.

We perform evaluation of two subset of KITTI dataset: KITTI stereo and KITTI VO, which both consist of challenging and varied road scene imagery collected from a test vehicle. Ground truth depth maps are obtained from 64-line LIDAR data. The main difference between these two dataset is that the ground truth depth maps for the stereo dataset were manually corrected and interpolated based on the neighboring frames. Note we only used the lower half part of the images (200×1226) as the upper half part generally include large part of sky and there is no depth measurements available. The input of our experiment was synthesized low-resolution LIDAR points which was downsampled by a factor of 6 in horizontal direction and 3 in vertical direction.

Piece-wise planar method setting: As this method performs on superpixel level, we use SLIC [132] segmentation method to provide segments and the number of superpixels is manually set to 800. The number of particles is the set to 10. The total iteration of PCBP is set to 40 as in Fig. 3.6 as the energies - both total and unary one - become flat since then.

"Cardboard world" method setting: We use SLIC [132] segmentation method to provide segments and the number of super-pixels is manually set to 1200. In the



(a) Unary energy versus the number of iterations.

(b) Total energy versus the number of iterations.

Figure 3.6: This figure shows the energy with respect to the number of iterations in CRF based method.

TRW-s inference step, the number of particles is set to 10, and divided into 3 parts: 1 particle represents the road plane, 4 particles are sampled from its neighbors and 5 particles are newly generated from MCMC process. The number of iteration of PCBP is set to 20 as there is only 1 parameter that need to optimize in this case.

3.3.2 Error Metrics

To evaluate the performance of depth interpolation, we use the following three quantitative metrics:

1) **Mean relative error** (MRE), which is defined as:

$$\mathrm{e}_{MRE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|\mathrm{d}_i - \widehat{\mathrm{d}}_i|}{\mathrm{d}_i}, \tag{3.7}$$

where d_i and $\widehat{\mathrm{d}}_i$ are the ground truth depth and inferred depth respectively. A lower MRE indicates a better dense depth prediction performance.

2) **Bad pixel ratio** (BPR) measures the percentage of erroneous positions in total, where a depth prediction result is determined as erroneous if the absolute depth prediction error is beyond a given threshold d_{th}. In our experiment, we set the bad pixel threshold as $\mathrm{d}_{th} = 3$ meters in VO dataset. A lower bad pixel ratio indicates a better depth prediction results.

3) **Mean absolute error** (MAE) is defined as:

$$\mathrm{e}_{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\mathrm{d}_i - \widehat{\mathrm{d}}_i|, \tag{3.8}$$

where d_i and $\widehat{\mathrm{d}}_i$ are the ground truth depth and depth prediction respectively. A

lower mean absolute error indicates a better dense depth prediction performance achieved. It also indicates the average depth estimation error in meters.

Bad pixel ratio, mean relative error and mean absolute error measure different statistics of the dense depth prediction results, which jointly evaluate the prediction performance.

3.3.3 Experiment Results

Our quantitative results are shown in Table 3.1. The piece-wise planar method outperforms all the other methods. However, since our goal is to generate both useful and visual pleasant 3D point cloud, we also provide qualitative results as shown in Fig. 3.7. For better comparison, we compare our method with several other state-of-the-art depth super-resolution methods: bilateral solver[32], as well as our colour-guided PCA based depth interpolation method. The initial depth map used for the bilateral solver [32] was generated by a general smooth interpolation method [129].

Table 3.1: Evaluation on the KITTI VO dataset.

	Bilateral[32]	colour PCA	Piece-wise	"Cardboard"
MRE(%)	7.36	5.73	**4.87**	7.85
BPR(%)	9.46	7.51	**5.82**	8.57
MAE(m)	1.20	1.04	**0.80**	1.29

As we can observe, bilateral solver [32] provides over-smoothed results, large distortion can be observed in the areas with different colours. For example, in Fig. 3.7, when there are shadows on road, it tends to assign same depth to same colour areas when lacking information, therefore create stripes effects in 3D point clouds. However, in our PCA based colour guided method, with the help of high order smoothness term and the PCA bases as a global constrain, it shows some resistances to the misleading of false boundaries that introduced by the colour images. Also, it can recover the shape of cars. However, as it is a pixel-wise algorithm, it is hard to estimate every pixel with the right depth. Therefore, we can find there are holes on the road plane. This can be harmful for autonomous driving system as it creates many false road pits and/or false obstacles.

The "cardboard world" algorithm, on the other hand, does not have these drawbacks. As we can see from all these results, none of the road plane was fooled by shadows or marks. For better illustration of our algorithm, in Fig. 3.8, we provide the input and output of our algorithm and the road plane segmentation as well. From top to bottom, each figure consists of the reference colour image, the input of our algorithm (LIDAR measurements and super-pixel segmentation), our recovered dense depth map and our road plane segmentation result correspondingly. *Note that the colour image is only used to generate the super-pixel segmentation and only sparse LIDAR measurements are used in the optimization.*

As we can see from the figures, all dominant road plane space (labeled by dark green) have been accurately extracted. In typical city road scenarios (i.e., Fig. 3.8), our

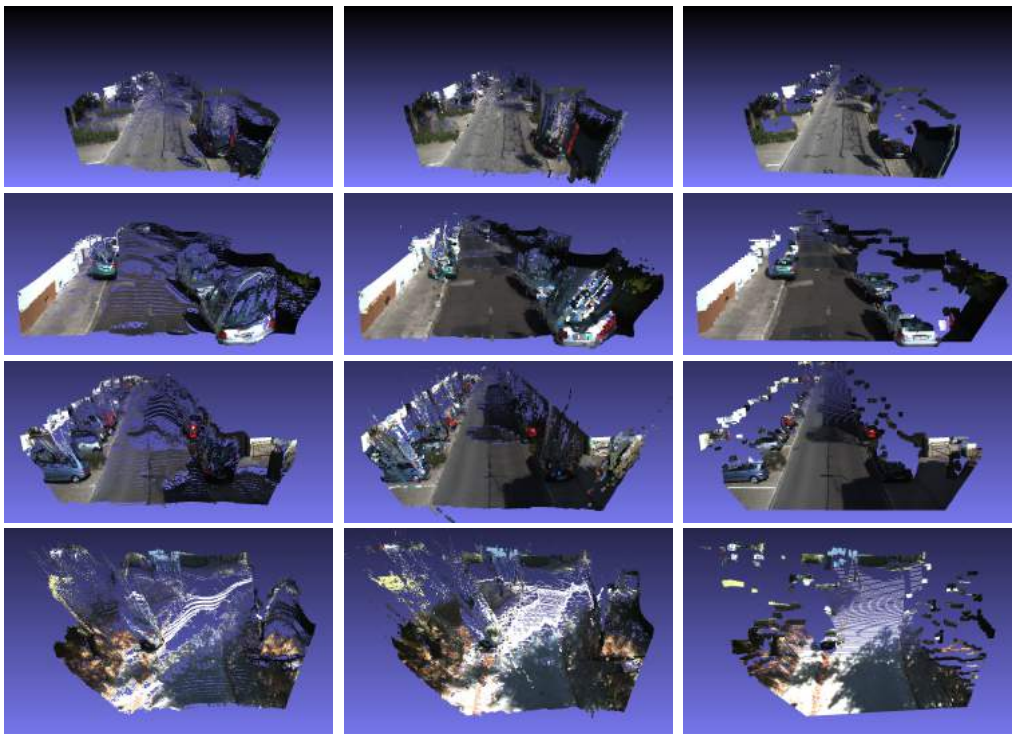


Figure 3.7: **Comparison in coloured 3D point clouds.** Left column: results from [32]; Middle column: results from PCA based colour guide method; Right column: results from the "cardboard world" method.

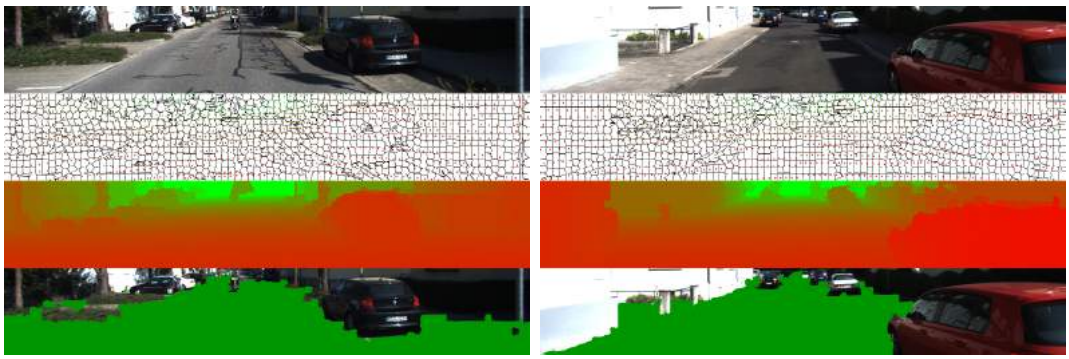
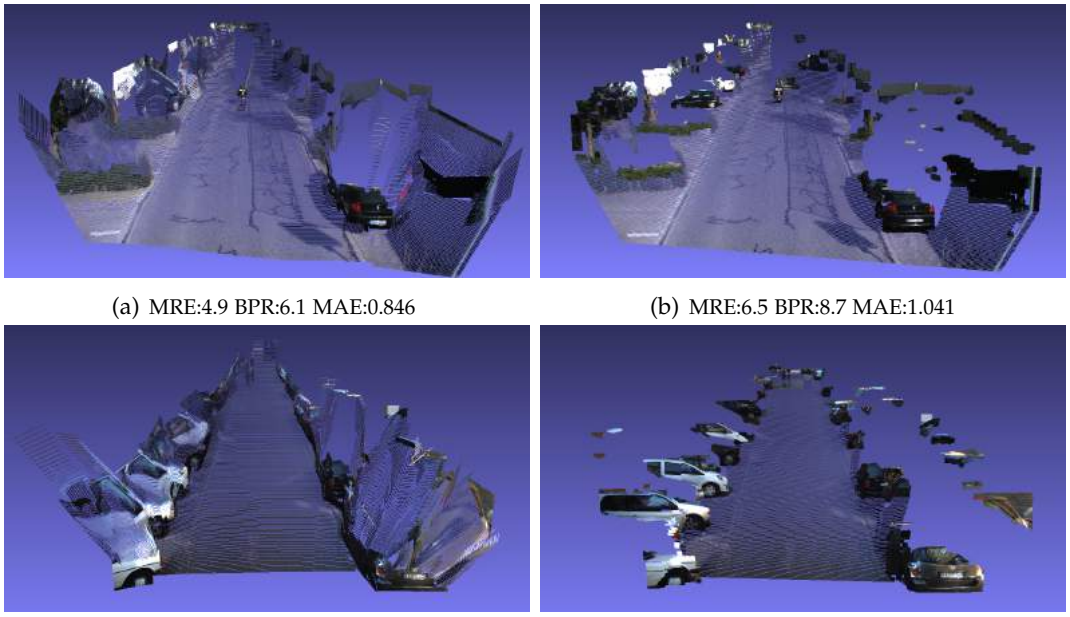


Figure 3.8: **Example results of "Cardboard world" method:** Top to bottom: reference colour frames, inputs of our method, recovered depth map and segmented free space (coloured with green).

method can successfully extract cars that are parked on side road and a motorbike over 22 meters away from only 3 LIDAR points on it.

To better illustrate the advantage of our "cardboard world" model over the stan-



(a) MRE:4.9 BPR:6.1 MAE:0.846 (b) MRE:6.5 BPR:8.7 MAE:1.041

(c) MRE:5.2 BPR:4.2 MAE:0.583 (d) MRE:9.5 BPR:6.2 MAE:1.060

Figure 3.9: Comparison between CRF based method (left) and "cardboard world" method (right).

dard piece-wise planar model, we also provided quality comparison between them. As we can see from Fig. 3.9, both methods successfully recover road plane with resistant to shadows. The quantitative results show that the CRF based method achieves better performance. However, in turns of the quality of 3D point cloud, the CRF method largely distorts the shape of cars as in (a) and (g) while the "cardboard world" method provides less distorted, much clean and visually look better results.

3.3.4 Processing time

Unlike pixel-level colour guided methods where the processing time largely depends on the input image resolution, the processing time of our method depends on the number of superpixels, the number of particles and the number of parameters to optimize. The higher the numbers, the longer the processing time is. In piece-wise planar model based solution, when we use 800 superpixels, 10 particles for each superpixel, the running time is about 10 seconds on average. However, in the more constrained case, our "cardboard world" method with more superpixels only needs 1 seconds on average.

3.4 Conclusion

This chapter aims at tackling the challenging task of predicting dense depth maps from very sparse measurements, we have proposed two different methods by exploits various local and global constraints inside the problem. The first method is based on the piecewise planar model of the scene, where dense depth prediction is reformulated as the optimization of planar parameters. The second method enforces strong regularization on the scene model to exploit the structural information in outdoor traffic scenes, which is more suitable for autonomous driving tasks. Unlike existing depth super-resolution methods that can be easily misled by marks or shadows on road, our methods inherently resist to these false guidance. Experimental results on the KITTI VO dataset show that our methods can efficiently recover dense depth map from less than 1000 LIDAR points without losing important information for autonomous driving, i.e., obstacles on road, or creating false obstacles that may misleading self-driving vehicles. In future, we plan to exploit the temporal information in constraining the dense depth maps.

Part III

Learning depth from a stereo camera

Chapter 4

Self-supervised Stereo Matching

Deep Learning based stereo matching methods have shown great successes and achieved top scores across different benchmarks. However, like most data-driven methods, existing deep stereo matching networks suffer from some well-known drawbacks such as requiring large amount of labeled training data, and that their performances are fundamentally limited by the generalization ability. In this chapter, we propose a novel Recurrent Neural Network (RNN) that takes a continuous (possibly previously unseen) stereo video as input, and directly predicts a depth-map at each frame without a pre-training process, and without the need of ground-truth depth-maps as supervision. Thanks to the recurrent nature (provided by two convolutional-LSTM blocks), our network is able to memorize and learn from its past experiences, and modify its inner parameters (network weights) to adapt to previously unseen or unfamiliar environments. This suggests a remarkable generalization ability of the net, making it applicable in an *open world* setting. Our method works robustly with changes in scene content, image statistics, and lighting and season conditions *etc*. By extensive experiments, we demonstrate that the proposed method seamlessly adapts between different scenarios. Equally important, in terms of the stereo matching accuracy, it outperforms state-of-the-art deep stereo approaches on standard benchmark datasets such as KITTI and Middlebury stereo [1].

4.1 Introduction

Stereo matching is a classic problem in computer vision, and it has been extensively studied in the literature for decades. Recently, deep learning based stereo matching methods are taking over, becoming one of the best performing approaches. As an evidence, they occupy the leader-boards for almost all the standard stereo matching benchmarks (*e.g.,* KITTI[89], Middlebury stereo [43]).

However, there exists a considerable gap between the success of these "deep stereo matching methods" on somewhat artificially created benchmark datasets and their real-world performances when being employed "in the wild" (open world), probably for the following reasons:

[1] This work was originally published in [15]

(1) Most of the existing deep stereo matching methods are supervised learning based methods, for which the training process demands massive annotated training samples. In the context of stereo matching, getting large amount of training data (*i.e.,* ground-truth disparity/depth maps) is an extremely expensive task.

(2) The performance of existing deep stereo matching methods and their applicability in real-world scenarios are fundamentally limited by their generalization ability: like most data-driven methods, they only work well on testing data that are sufficiently similar to the training data. Take autonomous driving for example, a deep stereo matching network trained in one city, under one traffic condition, might not work well in another city, under different lighting conditions.

(3) So far, most deep stereo matching methods exclusively focus on processing single pair of stereo images in a frame-by-frame manner, while in real world stereo camera captures continuous video. The rich temporal information contained in the stereo video has not been exploited to improve the stereo matching performance or robustness.

In this chapter, we tackle all the above drawbacks with current deep stereo matching methods. We propose a novel deep Recurrent Neural Network (RNN) that computes a depth/disparity map continuously from stereo video, without any pre-training process. Contrary to conventional stereo matching methods (*e.g.,* [7, 56]) which focus on processing a single pair of stereo images individually, this work is capitalized on explicitly exploiting the temporally dynamic nature of stereo video input.

Our deep stereo video matching network, termed as "OpenStereoNet" is not fixed, but changes its inner parameters continuously as long as new stereo frames being fed into the network. This enables our network to adapt to changing situations (*e.g.,* changing lighting condition, changing image contents, *etc.*), allowing it to work in unconstrained *open world* environments. OpenStereoNet is made of a convolutional Feature-Net for feature extraction, a Match-Net for depth prediction, and two recurrent Long Short-Term Memory (LSTM) blocks to encode and to exploit temporal dynamics in the video. Importantly and in contrast to existing deep stereo matching methods, our network does not need any ground-truth disparity map as supervision, yet it naturally generalizes well to unseen datasets. As new videos are processed, the network is able to memorize, and to learn from, its past experiences. Without needing ground-truth disparity maps, our network is able to tune its parameters after seeing more images from stereo videos, simply by minimizing image-domain warping errors. Also, to better leverage the sequential information in the stereo video, we apply the Long Short-Term Memory (LSTM) module to the bottleneck of feature extraction and feature matching part of our network. In the later part of this chapter, we demonstrate that our method can be applied to vary open-world scenarios such as indoor/outdoor scenes, different weather/light conditions and different camera settings with superior performance. Also ablation study concerning the effect of the LSTM modules is conducted. Another novelty of this work is that:

we adopt convolutional-LSTM [135] (cLSTM) as the recurrent feedback module, and use it directly on a continuous video sequence harnessing the temporal dynamics of the video. To our knowledge, while RNN-LSTM has been applied to other video processing tasks (such as sequence captions, or human action recognition), it has not been used for stereo matching for video sequences.

4.2 Related work

Stereo matching is a classic problem in computer vision, and has been researched for several decades. There have been significant number of papers published on this topic (The reader is referred to some survey papers *e.g.,* [43, 136]). Below we only cite a few most recent deep-learning based stereo methods that we consider most closely related to the method to be described.

Supervised Deep Stereo Matching. In this category, a deep network (often based on CNN, or Convolutional Neural Networks) is often trained to benefit the task of stereo matching in one of the following aspects: i) to learn better image features and a tailored stereo matching metrics (*e.g.,* [47, 137]); ii) to learn better regularization terms in a loss function [48]; and iii) to predict dense disparity map in an end-to-end fashion (*e.g.,* [51, 56]). The learned deep features replace handcrafted features, resulting in more distinctive features for matching. End-to-end deep stereo methods often formulate the task as either depth values regression, or multiple (discrete) class classification. DispNetC [51] is a new development, which directly computes the correspondence field between stereo images by minimizing a regression loss. Another example is the GC-Net [56], which explicitly learns feature cost volume, and regularization function in a network structure. Cascade residual learning (CRL) [138] adopted a multi-stage cascade CNN architecture, following a coarse-to-fine or residual learning principle [139].

Unsupervised Deep Stereo Matching. Recently, there have been proposed deep net based single-image depth recovery methods which do not require ground-truth depth maps. Instead, they rely on minimizing photometric warping error to drive the network in an unsupervised way (see *e.g.,* [112, 109, 97, 113, 14]). Zhou *et al.* [97] proposed an unsupervised method which is iteratively trained via warping error propagating matches. The authors adopted TV (total variation) constraint to select training data and discard uninformative patches. Inspired by recent advances in direct visual odometry (DVO), Wang *et al.* [114] argued that the depth CNN predictor can be learned without a pose CNN predictor. Luo *et al.* [115] reformulated the problem of monocular depth estimation as two sub-problems, namely a view synthesis procedure followed by standard stereo matching. However, extending these monocular methods to stereo matching is non-trivial. When feeding the networks with stereo pairs, their performances are even not comparable with traditional stereo matching methods [109].

Recurrent Neural Net and LSTM. Our method is based on RNN (with cLSTM as the feedback module), and directly applied to sequence input of stereo video

harnessing the temporal dynamic nature of a continuous video. To the best of our knowledge, where RNN-LSTM has been applied to other video based tasks (such as a sequence captions, action recognition), it has not been directly used for stereo video matching, especially to exploit the temporal smoothness feature for improving stereo matching performance.

4.3 Network Architecture

In this section, we describe our new "open-world" stereo video matching deep neural network (for ease of reference, we call it *OpenStereoNet*). The input to the network is a live continuous stereo video sequence of left and right image frames of I_L^t,I_R^t, for $t = 1, 2, ...$ The output is the predicted depth-map (disparity map) at each time step t. We assume the input stereo images are already rectified.

Our network does not require ground-truth depth-maps as supervision. Instead, the stereo matching task is implemented by searching a better depth map which results in minimal photometric warping error between the stereo image pair. By continuously feeding in new stereo image frames, our network is able to automatically adapt itself to new inputs (could be new visual scenes never seen before) and produce accurate depth map estimations. More technical details will be explained in the sequel of the paper.

4.3.1 Overall network architecture

The overall structure of our OpenStereoNet is illustrated in Figure-4.1, which consists of the following major parts (or sub-Nets): (1) Feature-Net, (2) Match-Net, (3) LSTM blocks, and (4) a loss function block.

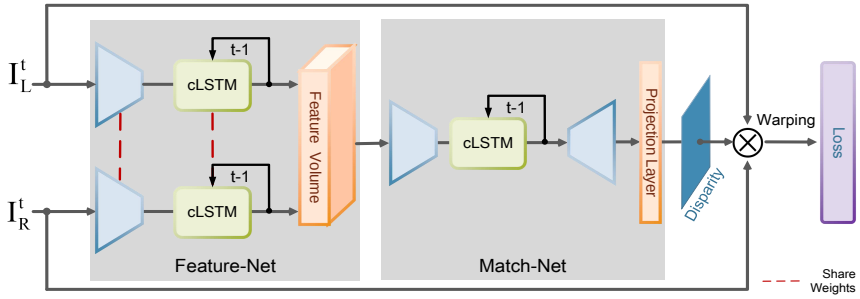


Figure 4.1: **Overall network structure of our OpenStereoNet.** It consists of a convolutional Feature-Net, an encoder-decoder type Match-Net and two recurrent (convolutional) LSTM units to learn temporal dynamics of the video input. Given a stereo pair I_L^t, I_R^t at time t, the Feature-Net produces feature maps which are subsequently aggregated to form a feature-volume. The Match-Net first learns a representation of the feature volume then projects it to obtain disparity estimation. Our loss function is based on image warping error evaluated on raw image inputs by the current disparity map.

Information Flow. Starting from inputted left and right images at time t, the information processing flow in our network is clear: 1) The Feature-Net acts as a convolutional feature extractor which extracts features from the left and right images individually. Note, Feature-Net for the left image and Feature-Net for the right image share the weights. 2) The obtained feature maps are concatenated (with certain interleave pattern) into a 4D feature-volume. 3) The Match-Net takes the 4D feature volume as input, and learns an encoder-decoder representation of the features. A projection layer (based on soft-argmin [56]) within the Match-Net is applied to produce the 2D disparity map prediction. Finally, the loss function block employs the current estimated disparity map to warp the right image to the left view and compare the photometric warping loss as well as other regularization term, which is used to refine the network via *backprop*.

4.3.2 Feature-Net

Conventional stereo matching methods often directly compare the raw pixel values in the left image with that in the right image. Recent advance in deep learning show that using learned convolutional features can be more robust for various vision tasks. For stereo matching, to learn a feature map that is more robust to photometric variations (such as occlusion, non-lambertian, lighting effects and perspective effects) will be highly desirable.

In this chapter, we design a very simple convolutional feature-net with 18 convolutional layers (including RELU) using 3×3 kernels and skip connections in between. The output feature has a dimensionality of 32. We run feature extraction on both images in a symmetric weight-sharing manner.

4.3.3 Feature-Volume construction

We use the learned features to construct a feature volume. Instead of constructing a cost volume by concatenating all costs with their corresponding disparities, we concatenate the learned features from the left and right images. Specifically, we concatenate each learned feature with their corresponding feature from the opposite stereo image across each disparity level in a preset disparity range D as illustrated in Fig. 4.2. All the features are packed to form a 4D feature volume with dimensionality $H \times W \times (D+1) \times 2F$ for the left-to-right and right-to-left feature volume correspondingly, where H, W, D, F represent the height, width, disparity range, and feature dimensionality respectively.

4.3.4 Match-Net

Taking the assembled Feature-Volume as input, our Match-Net is constituted of an encoder-decoder as the front-end, followed by a single last layer which projects the output of the encoder-decoder to a 2D disparity-map.

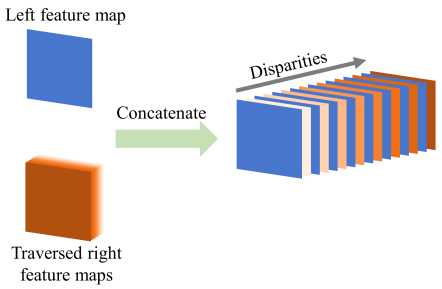


Figure 4.2: **Feature Volume Construction.** We collect the two feature maps computed by the Feature-Net, and assemble them together to a Feature-Volume in the way as illustrated here: the blue rectangle represents a feature map from the left image, the stacked orange rectangles represent traversed the right feature at different disparities in the range $[0, D]$. Note that the left feature map is duplicated $D + 1$ times to match the traversed right feature maps.

4.3.4.1 Encoder-Decoder front-end.

The denoising Encoder-Decoder is an hour-glass-shaped deep-net. Between the encoder and decoder there is a bottleneck, as shown in Figure-4.1. Since the input feature-volume is of 4 dimensions, H (height) $\times$ W (width) $\times$ (D+1)(disparity range) $\times$ 2F(feature dim.), we use 3D-convolutional kernels and the underlying CNNs in the Encoder-Decoder are in fact 3D-CNNs.

4.3.4.2 Projection layer.

The output of the preceding encoder-decoder is still a 4D feature-volume. The last layer of our Match-Net first projects the 4D volume to a 3D *cost volume–i.e.,* an operation commonly used in conventional stereo matching methods, then applies the soft-argmin operation (*cf.* [56]) to predict a disparity $\delta = \sum_{d=0}^{D}[d \times \sigma(-c_d)]$, where c_d is the matching cost at disparity d and $\sigma(\cdot)$ denotes the softmax operator.

4.3.5 Convolutional-LSTM

Since our goal is to develop a deep-net focusing on stereo video processing (as opposed to individual images), in order to capture the inherent temporal dynamics (*e.g.,* temporal smoothness) that exist in a video, we leverage the internal representations obtained by our two sub-networks (*i.e.,* , Feature-Net and Match-Net), and model these internal representation's dynamic transitions as an *implicit* model for the video sequence. Specifically, given a continuous video, we consider the image content (as well as the disparity) in each frame changes smoothly to the next frame. To capture such dynamic changes, we adopt the structure of LSTM-based Recurrent Neural Networks (RNN). The LSTMs act as memory of the net, by which the network memorizes its past experiences. This gives our network the ability to learn the stereo video sequence's temporal dynamics encoded in the inner states of the LSTMs. As shown in Fig.-4.3, the output of our Feature-Net, and the encoder-decoder in our

Match-Net, are each passed to an LSTM unit. Briefly, LSTM units are a particular type of hidden unit that improve the training of RNNs [140]. An LSTM unit contains a cell, which can be thought of as a memory state. Access to the cell is controlled through an input gate and a forget gate. The final output of the LSTM unit is a function of the cell state and an output gate (*cf.* [141]).

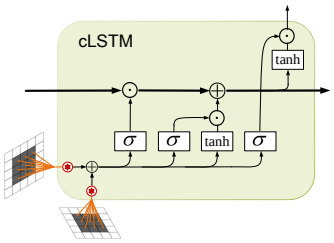


Figure 4.3: **A convolutional-LSTM.**

We realize that for stereo matching it is desirable to use spatially-invariant operator such as convolutional kernels, in the same spirit of the CNN. In light of this, we propose to use the convolutional LSTM architecture (or cLSTM in short, *cf.* [135]) as the recurrent unit for our stereo video matching task. This way, the relative spatial layout information of the feature representation–which is essential for the task of depth map computation–is preserved. In our experiments we used very small kernels for the cLSTM (*e.g.*, 3×3, or 5×5). This leads to compact LSTM units, and greatly simplifies the computation cost and GPU-RAM consumption. The encoder–LSTM–decoder architecture is also similar to the Encoder-Recurrent-Decoder architecture proposed in [142]. Our entire network works end-to-end to combine feature representation learning with the learning of temporal dynamics via the two LSTM blocks.

4.4 Self-Adapting Learning and Loss Function

4.4.1 Self-Adapting Training and Testing

Recall that the ultimate goal of this work is to develop a deep network that can automatically adapt itself to new (previously unseen) stereo video inputs. In this sense, the network is not fixed static, but is able to dynamically evolve in time. This is achieved by two mechanisms:

- The network has memory units (i.e. the two LSTM blocks), which enable the network to adjust its current behavior (partly) based its past experiences;
- We always run an online back-propagation (*backprop*) updating procedure after any feed-forward process.

The latter actually eliminates the separation between a network's training stage and testing stage. In other words, our OpenStereoNet is constantly performing both operations all the time. This gives the network self-adaption ability, allows it to self-adapt by continuously fine-tuning its parameters based on new stereo image inputs (possibly seen in a new environment). Thus, our OpenStereoNet can "automatically" generalize to unseen images.

Since we do not require ground-truth depth-maps as supervision, input stereo pairs themselves serve as self-supervision signals, and the network is able to update automatically, by self-adapting learning.

4.4.2 Overall loss function

The overall loss function for our OpenStereoNet is a weighted summation of a data term and a regularization term, as in $Loss = L_{\text{data}} + \mu L_{\text{reg}}$.

Data term: Image warping error. We directly measure the warping error evaluated on the input stereo images, based on the estimated disparity map. Specifically, given the left image I_L^t and the disparity map for the right image $d_R^t = g(I_R^t, I_L^t, h_R^{t-1}, h_L^{t-1})$, the right image I_R^t can be reconstructed by warping the left image with d_R^t, $I_R^{t'}(u,v) = I_L^t(u+d,v)$, where $I_R^{t'}$ is the warped right image. We use the discrepancy between $I_R^{t'}$ and the observed right image I_R^t as the supervision signal. Our data loss is derived as: $L_{\text{data}} = \sum(\lambda_1(1 - \mathcal{S}(I_L, I_L^{'}))/2 + \lambda_2(\left|I_L - I_L^{'}\right| + \left|\nabla I_L - \nabla I_L^{'}\right|))/N$. The data term consists of S, which is the structural similarity SSIM as defined in [143], pixel value difference and image gradient difference. The trade-off parameters were chosen empirically in our experiments at $\mu = 0.05, \lambda_1 = 0.8, \lambda_2 = 0.1$.

Regularization term: Priors on depth-map. We enforce a common prior that depth-maps are piecewise smooth or piecewise linear. This is implemented by penalizing the second-order derivative of the estimated disparity map. To exploit correlations between depth map values and pixel colours, we weight this term by image colour gradient, *i.e.*, : $L_{\text{reg}} = \sum(e^{-\left|\nabla_u^2 I_L\right|}\left|\nabla_u^2 d_L\right| + e^{-\left|\nabla_v^2 I_L\right|}\left|\nabla_v^2 d_L\right|)/N$, where ∇ is gradient operator.

4.5 Experiments

We implement our OpenStereoNet in TensorFlow. Since the network runs in an online fashion (with batch-size one), *i.e.*, , there is no clear distinction between training and testing, we start from randomly initialized weights for both the Feature-Net and the Match-Net, and allow the network to evolve as new stereo images being fed in. All images have been rescaled to 256×512 for easy comparison. Typical processing time of our net is about 0.8–1.6 seconds per frame tested on a regular PC of 2017 equipped with a GTX 1080Ti GPU. We use the RMSProp optimizer with a constant learning rate of 0.001. We have evaluated our network on several standard benchmark datasets for stereo matching, including KITTI[89], Middlebury[144, 145], Synthia [146], and Frieburg SceneFlow [51] (e.g. FlyingThings3D). These experiments are reported below.

4.5.1 KITTI visual odometry (VO) stereo sequences

In this set of experiments on KITTI dataset [89], we simply feed a KITTI VO stereo video sequence to our network, and start to produce a depth map prediction, as well as update the network weights frame by frame by backproping the error signal of the loss function. In all our experiments we observe that: soon after about a few hundreds of input frames have been processed (usually about 10 seconds video

at 30fps) the network already starts to produce sensible depth maps, and the loss function appears to converge. We call this process of training on the first a few hundred frames the *network prime* process, and we believe its purpose is to teach the network to learn useful convolutional features for typical visual scenes. Once the network has been "primed", it can be applied to new previously unseen stereo videos.

Figure-4.4 shows typical converge curves for a network during the prime stage. After the prime stage, we randomly select 5 KITTI VO sequences, test our network

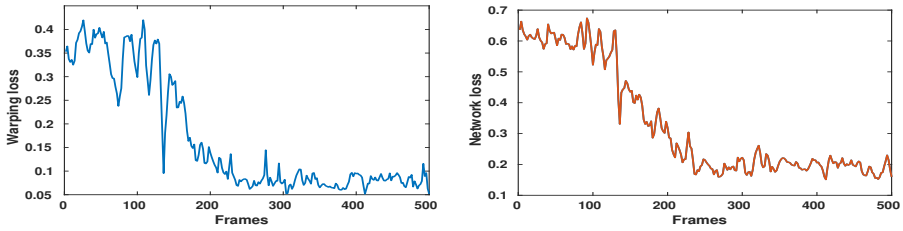
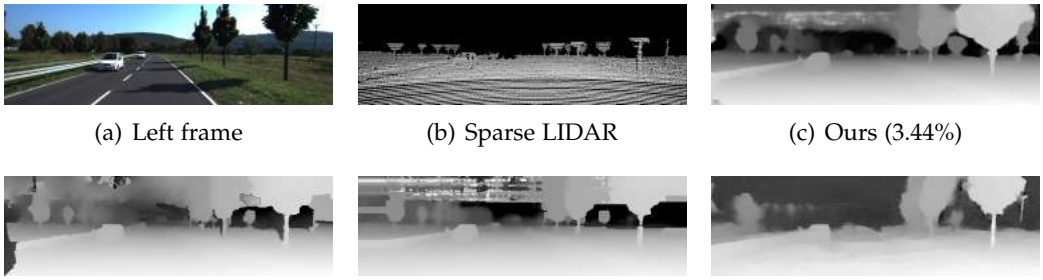


Figure 4.4: **Typical network convergence curves from random initialization.**

on them, and compare its performance with three state-of-the-art stereo methods, including the DispNet [51], MC-CNN [47], and SPS-ST [131]. The first two are deep stereo matching methods, and the last one a traditional (non-deep) stereo method. Quantitative comparison of their performances are reported in Table-4.1, from which one can clearly see that our OpenStereoNet achieves the best performance throughout all the metrics evaluated. For deep MC-CNN we use a model which was firstly trained on Middlebury dataset for the sake of fair comparison. For SPS-ST, its meta-parameters was also tuned on KITTI dataset. Figure–4.5 gives some sample visual



(a) Left frame (b) Sparse LIDAR (c) Ours (3.44%)

(d) SPS-st (5.61%) (e) MC-CNN (4.75%) (f) DispNet (25.98%)

Figure 4.5: **Our qualitative results on KITTI VO dataset:** Results are reported on the *D1_all* error metric. Disparities are transformed to log space for better visualization.

results for comparison. Note that our method obtains sharp and clean depth discontinuity for cars and trees, better than the other methods. Dispnet, on the other hand, is affected by shadows on road, with many artifacts on the road surface. A quantitative comparison is provided in Table-4.1. Our method outperforms all baseline

methods with a large margin.

Table 4.1: Quantitative results on KITTI VO dataset.

Methods	Abs Rel	Sq Rel	RMSE	RMSE log	D1_all	$\delta < 1.25$
Dispnet [51]	0.122	1.938	8.844	0.189	32.045	0.877
MC-CNN [47]	0.069	1.229	6.002	0.264	8.018	0.932
SPS-st [131]	0.060	1.341	5.521	0.159	4.970	0.957
Ours	**0.053**	**0.540**	**4.451**	**0.137**	**4.403**	**0.959**

4.5.2 Synthia Dataset

The Synthia dataset [146] contains 7 sequences with different scenarios under different seasons and lighting conditions. Our primary aim for experimenting on Synthia is to analyze our network's generalization (self-adaption) ability. We create a long video sequence by combining together three Synthia sequences of the same scene but under different seasons and lighting conditions. For example, Fig.-4.6 shows some sample frames of *Spring, Dawn and Night*. We simply run our network model on this video, and display the disparity error as a function of frames. We run our network on this long sequence. For each condition, we report our quantitative and qualitative results based on the first 250 frames of that sequence.

As shown in Fig. 4.6, our network recovers consistently high quality disparity maps regardless the lighting conditions. This claim is further proved by the qualitative results in Fig.-4.7. In term of disparity accuracy, our method achieves a Mean Absolute Error (MAE) of 0.958 pixels on the *Spring* scene while the *Dawn* sequence has reached an MAE of 0.7991 pixels and 1.2415 pixels for the *Night* sequence.

4.5.3 Ablation Studies: Effects of the LSTMs and Backprop

There are two mechanisms that contribute to the self-adaptive ability of our Open-StereoNet, *i.e.*, , the cLSTMs recurrent blocks and the backprop refinement process. To understand their respective effects on the final performance of our network, we conduct ablation studies by isolating their operations. To be precise, we have tested the following four types of variants of our full networks: (type-1) remove LSTMs and also disable the backprop process (*i.e.*, , the *baseline* network); (type-2) remove LSTMs, but keep backprop on; (type-3) with LSTMs on, without backprop, and (type-4) with both LSTMs on and backprop on (*i.e.*, , our full network).

Results by these four types of networks are given in the following curves in Figure-4.8. One can clearly see the positive effects of the LSTM units and the backprop. In particular, adding LSTMs has reduced the loss function of the baseline network significantly.

In another ablation test, we run the above type-3 network on the previous selected KITTI VO sequence, and list their accuracy in Table- 4.2. From this, one can see that by applying the LSTM module, we have achieved better performance across all error metrics.

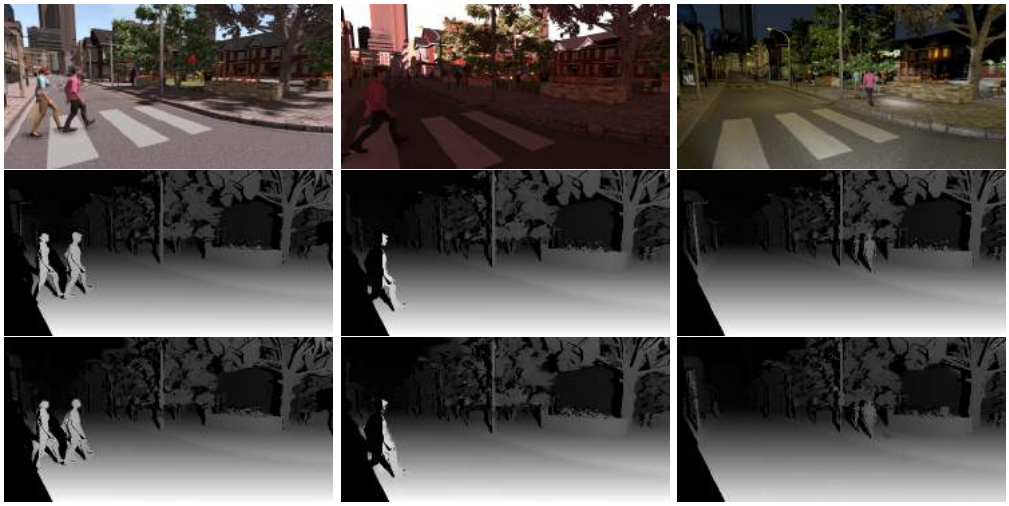


Figure 4.6: **Our qualitative results on Synthia:** Top to bottom: input left image, ground truth disparity, our result. The first column is taken from the *Spring* subset, the middle column is from *Dawn*, and the last column is from *Night*. Our method performs uniformly on different sequences.

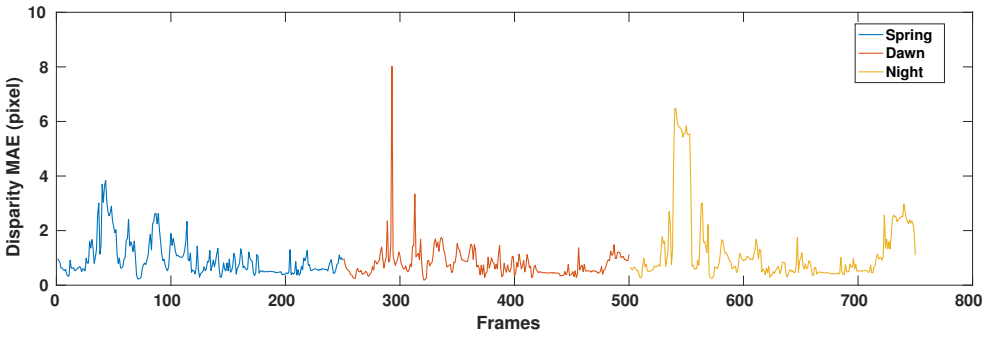


Figure 4.7: We run our method on a continuous video sequence consisting of the same scene under three different season/lighting conditions (spring, dawn, night). The curve shows the final disparity error as a function of the stereo frame. From this curve it is clear that our network is able to adapt to new scenarios automatically.

Table 4.2: Ablation study on LSTM module on KITTI.

Methods	Abs Rel	Sq Rel	RMSE	RMSE log	D1_all	$\delta < 1.25$
Type-3 net (without LSTMs)	0.066	1.580	5.332	0.167	5.089	0.957
Type-4 net (with LSTMs)	**0.053**	**0.540**	**4.451**	**0.137**	**4.403**	**0.959**

4.5.4 Middlebury Stereo Dataset

The stereo pairs in the Middlebury stereo dataset [144, 145] are indoor scenes with multiple handcrafted layouts. The ground truth disparities are captured by structured light with higher density and precision than KITTI dataset. We report our

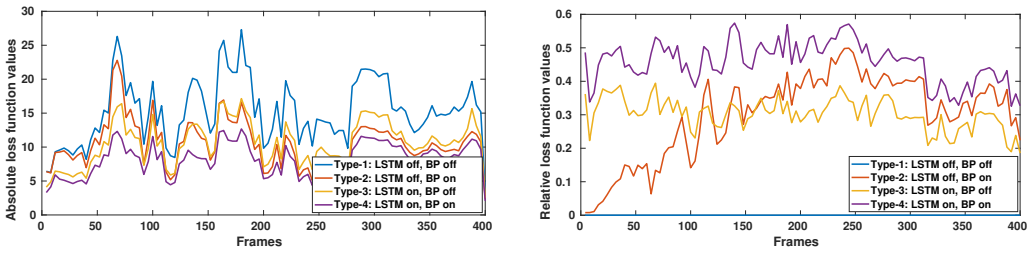
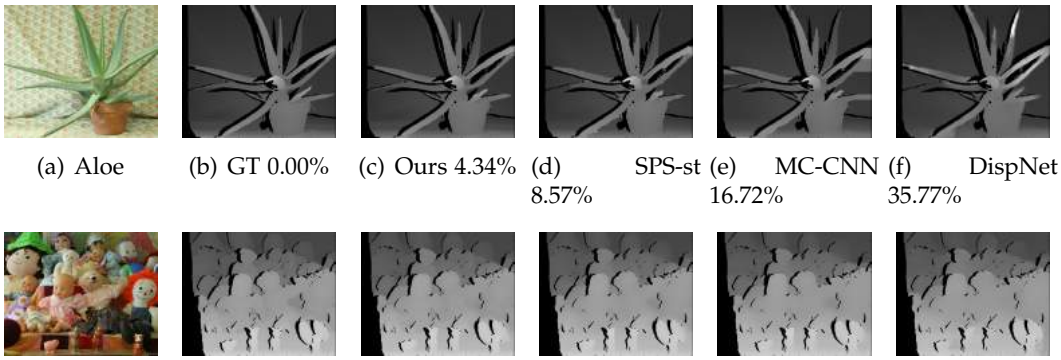


Figure 4.8: We run our method on a continuous video sequence. The left figure shows the comparison of absolute losses for the 4 types (note: the lower, the better), while the right figure gives comparison of the relative loss against the the (type-1) baseline network (note: the higher, the better). The y-axis indicates the final disparity errors, and x-axis the input frame-Ids.

results on selected stereo pairs from Middlebury 2005 [144] and 2006 [145] and compare with other baseline methods. In order to evaluate our method on these images, we augmented each stereo pair to a stereo video sequence by simply repeating the stereo pair.

We use *bad-pixel-ratio* as our error metrics used in this experiment, and all results are reported with 1-pixel thresholding. As shown in Fig. 4.9, our method achieves superior performance than all baseline methods. Other deep learning based methods have even worse performance than the conventional method SPS-st when there is no fine tuning.



(a) Aloe (b) GT 0.00% (c) Ours 4.34% (d) SPS-st 8.57% (e) MC-CNN 16.72% (f) DispNet 35.77%

(g) Dolls (h) GT 0.00% (i) Ours 6.88% (j) SPS-st 15.54% (k) MC-CNN 23.78% (l) DispNet 44.52%

Figure 4.9: **Our results on Middlebury:** Left to right: Left image, ground-truth disparity, our result, SPS-st result, MC-CNN result and DispNet result. We report the *bad-pixel-ratio* at 1-pixel threshold.

4.5.5 Other open world stereo sequences

To further demonstrate the generalization ability of our OpenStereoNet, we test it on a number of other freely downloaded stereo video datasets from the Internet. Note that our network had never seen these test data before. Below we give some sample results, obtained by our method and by the DispNet, on the Freiburg Sceneflow Dataset [51] and on RDS-Random Dot Stereo.

Freiburg Sceneflow Dataset. We select two stereo videos from the Monkaa and FlyingThings3D dataset [51] and directly feed them into our network. Qualitative results are shown in Figure-4.10 and Figure-4.11 correspondingly. Our network produces very accurate disparity maps when compared with the ground truth disparity maps. Furthermore, the reconstructed colour images with the estimated disparity map further prove the effectiveness of our model.

Random dot stereo. We test the behavior of our OpenStereoNet on random dot stereo images where there is no semantic content in the images. Our network works well, however the DispNet fails miserably as shown in Figure-4.12.

Figure 4.10: **Our qualitative results on Monkaa:** From top to bottom: Left image, reconstructed left image, recovered disparity map. From left to right: frame 11, frame 15, frame 133, frame 143.

4.6 Conclusions and Discussions

This chapter addresses a practical demand of deploying stereo matching technique to unconstrained real-world environments with previously unseen or unfamiliar "open-world" scenarios. We envisage such a stereo matching method that is able to take a continuous live stereo video as input, and automatically predict the corresponding disparity maps. To this end, this chapter has proposed a deep Recurrent Neural Network (RNN) based stereo video matching method–*OpenStereoNet*. It consists of a CNN Feature-Net, a Match-Net and two convolutional-LSTM recurrent blocks to

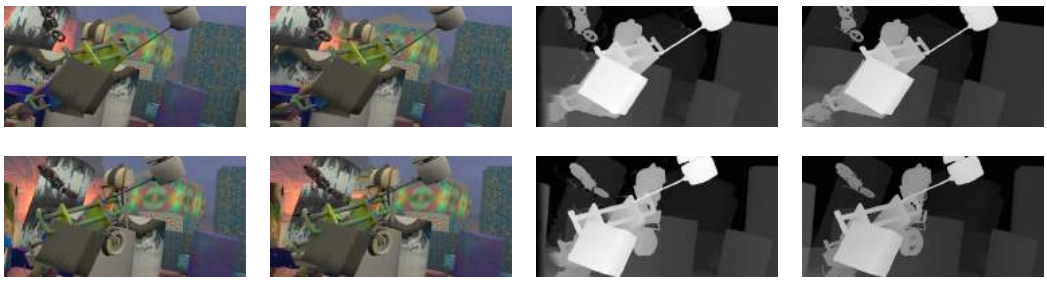


Figure 4.11: **Our qualitative results on FlyingThings3D:** From left to right: Left image, reconstructed left image, estimated disparity map, and ground truth.

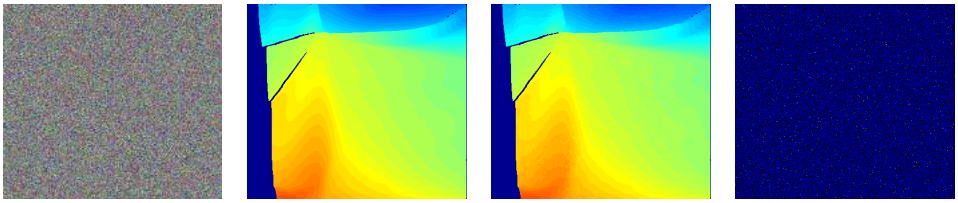


Figure 4.12: **Test results on RDS (Random dot stereo) images:** Left to right: input stereo (left image), ground-truth disparity (colour coded), our result, result by Dispnet. Our method successfully recovers the correct disparity map, demonstrating its superior generalibility on unseen images.

learn temporal dynamics in the scene. We do notice that finding optical flow (or scene flow) between image frames is yet another feasible paradigm to encode and to exploit temporal dynamics existed in a video sequence. However, we argue optical flow itself is a significant research topic in itself, no less challenging than stereo matching, and a comparison between the two approaches deserves to be a valuable future work.

Our OpenStereoNet does not need ground-truth disparity maps for training. In fact, there is even no clear distinction between training and testing as the network is able to learn on-the-fly, and to adapt itself to never-seen-before imageries rapidly. We have conducted extensive experiments on various datasets in order to validate the effectiveness of our network. Importantly, we have found that our network generalizes well to new scenarios. Evaluated based on absolute performance metrics for stereo, our method outperforms state-of-the-art competing methods by a clear margin.

Chapter 5

Displacement Invariant Cost Computation for Efficient Stereo Matching

Although deep learning-based methods have dominated stereo matching leaderboards with superior estimation accuracy, their inference time is long, typically in the order of seconds for a pair of 540p images. The main reason is that leading methods employ 3D convolutions applied to a 4D feature volume, which is very time consuming. A common strategy to speed up the computation is to downsample the feature volume, but this loses high-frequency details. To overcome these challenges, we propose an *efficient cost-aggregation module* to compute the matching costs without needing a 4D feature volume. Rather, costs are aggregated by applying the same 2D convolution network on each disparity-shifted feature map pair independently. Unlike previous 2D convolution-based methods that simply perform context mapping between inputs and disparity maps, our proposed approach learns to match features between the two images. We also propose an entropy-based refinement strategy to refine the computed disparity map, which further improves speed by avoiding the need to compute a second disparity map on the right image. Extensive experiments on standard datasets (SceneFlow, KITTI, ETH3D, and Middlebury) demonstrate that our method achieves competitive accuracy with much less inference time. On typical image sizes, our method processes over 100 FPS on a desktop GPU, making our method suitable for time-critical applications such as autonomous driving. We also show that our approach generalizes well to unseen datasets, outperforming 4D-volumetric methods[1].

5.1 Introduction

Deep learning-based methods have achieved state-of-the-art on most of the standard stereo matching benchmarks (*i.e.,* KITTI [90], ETH3D [148], and Middlebury [149]). This success is achieved by aggregating information in a 4D feature volume (height

[1] A similar work in optical flow was published in [147].

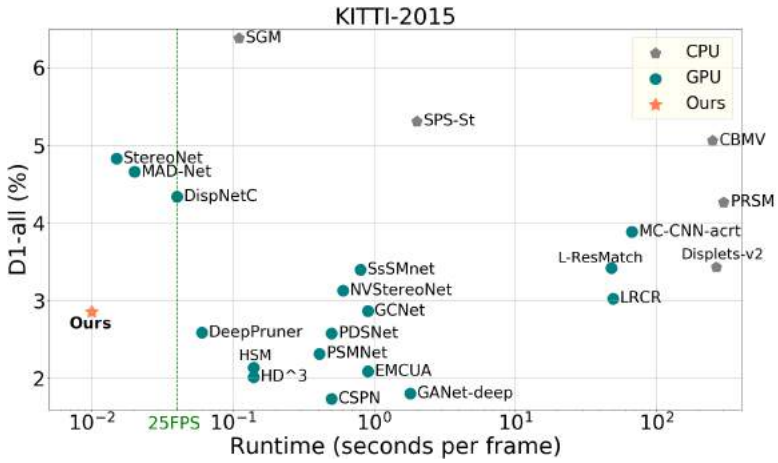


Figure 5.1: Our method achieves stereo matching accuracy comparable to state-of-the-art on the KITTI 2015 *test* dataset, while operating at 100 FPS. (Best viewed on screen.)

$\times$ width $\times$ disparity levels $\times$ feature dimension), which is formed by concatenating each feature in one image with its corresponding feature in the other image, across all pixels and disparity levels. To process such a 4D volume, expensive 3D convolutions are utilized, thus making these methods significantly more time- and space-intensive than traditional approaches like semi-global matching (SGM) [7] and its variants [150]. For example, the traditional method known as embedded SGM [150] achieves 100 FPS (frames per second) for a pair of 540p images, whereas most deep learning-based methods only manage about 2 FPS. Moreover, since this 4D feature volume grows with the cube of resolution, high-resolution depth estimation is prohibitively expensive.

In this chapter, we propose to overcome these limitations with an *efficient cost aggregation module* that learns to match features of stereo images *using only 2D convolutions*. Unlike previous 2D convolution-based methods [51, 52, 138], however, ours does not rely upon context matching between pixels and disparities; rather, due to its unique design, our network learns to match pixels between the two images. The key insight behind our approach is to compute the cost of each disparity shift independently using the same 2D convolution-based network. This vastly reduces the number of parameters and memory requirements for training and it also reduces time and memory costs during inference, as it does not need to explicitly store a 4D feature volume before cost aggregation. Also, we leverage entropy maps that computed from 3D cost volume as confidence maps to guide the refinement of disparities. As a result, our proposed method is not only significantly faster than 3D convolution-based volumetric methods, *e.g.*, GA-Net [50], but it also achieves better cross-dataset generalization ability. The entire system is trained end-to-end.

Our method contains the following contributions:

- An efficient cost aggregation module for stereo matching that uses 2D convolutions on disparity-shifted feature map pairs, which achieves significant speedup

over standard volumetric approaches with much lower memory requirements.
- A new entropy based refinement scheme that bypasses the need for estimating disparity for the right view, which further reduces the processing time and memory consumption.
- State-of-the-art performance on all three benchmarks compared with existing real-time methods, and better generalization performance than existing methods.

5.2 Related work

Given a pair of rectified stereo images, stereo matching attempts to find a matching point for each pixel on the corresponding epipolar line. Stereo matching has been extensively studied for decades in computer vision. Here we discuss a few popular and recent methods that are closely related to our approach. Interested readers are referred to recent survey papers such as [43] and [136].

5.2.1 Traditional Stereo Matching

Traditional stereo matching methods can be roughly divided into two classes: local methods and global methods. The typical pipeline for a local method has four consecutive steps: 1) compute costs at a given disparity for each pixel; 2) sum up the costs over a window; 3) select the disparity that has the minimal cost; 4) perform a series of post-processing steps to refine the final results. Local methods [43, 44] have the advantage of speed. Since each cost within a window can be independently computed, these methods are highly parallelizable. The disadvantage of such methods is that they can only achieve sub-optimal results because they only consider local information. Global methods [45, 46] have been proposed to address this issue. They treat the disparity assignment task as a maximum flow / minimum cut problem and try to minimize a global cost function. However, such algorithms are typically too slow for real-time applications. Semi-global matching (SGM) [7] is a compromise between these two extremes, which could achieve more accurate results than local methods without sacrificing speed significantly.

5.2.2 Deep Stereo Matching

Unlike traditional methods, deep stereo matching methods can learn to deal with difficult scenarios, such as repetitive textures or textureless regions, from ground truth data. A deep stereo method is often trained to benefit from one of the following aspects: 1) learning better features or metrics [47]; 2) learning better regularization terms [48]; 3) learning better refinement [49]; 4) learning better cost aggregation [50]; 5) learning direct disparity regression [51]. Based on the network structure, we divided these methods into two classes:

Direct Regression methods often use an encoder-decoder to directly regress disparity maps from input images [51, 52, 53, 54]. Such methods learn in a brute force man-

ner, discarding decades of acquired knowledge obtained by classical stereo matching research. When there are enough training data, and the train and test distributions are similar, such methods can work well. For example, iResNet [52] won first place in the 2018 Robust Vision Challenge. Also, since such methods only employ 2D convolutions, they can easily achieve real-time or near real-time processing and have low GPU memory consumption. However, these methods lack the ability to generalize effectively, *e.g.,* DispNet [51] fails random dot stereo tests [15].

Volumetric Methods [56, 50, 14, 57] build a 4D feature volume using features extracted from stereo pairs, which is then processed with multiple 3D convolutions. These methods leverage the concept of semi-global matching while replacing handcrafted cost computation and cost aggregation steps with 3D convolutions. Since this type of network is forced to learn matching, volumetric methods easily pass random dot stereo tests [15]. However, they suffer from high processing times and high GPU memory consumption [50]. To overcome these issues, some researchers build the feature volume at a lower image resolution to reduce memory footprint [49, 151], or prune the feature volume by generating a confidence range for each pixel and aggregating costs within it [152]. However, lowering the resolution of the feature volume makes it difficult to recover high-frequency information. To restore these lost details, volumetric networks typically use colour-guided refinement layers as a final processing stage.

5.2.3 Real-time Stereo Matching

Several real-time deep stereo matching networks have been proposed recently. StereoNet [49] achieves real-time performance at the cost of losing high frequency details due to a low resolution 4D cost volume, *i.e.,* 1/8 or 1/16 of the input images. DispNet [51] and MADNet [53] discard the general pipeline of stereo matching and design a context regression network that directly regresses disparity maps from the input images; these approaches suffer generalization problems as shown in [15].

To overcome the limitations of previous methods, our proposed approach uses an efficient cost aggregation module to achieve super real-time inference while, at the same time, preserving most high-frequency details by constructing a 3D cost volume on 1/3 of the input resolution. Moreover, experiments in Section 5.4.3 show that our method achieves better generalization ability than 4D volumetric methods.

5.3 Method

In this section, we first describe the overall architecture and then provide a detailed analysis of our proposed *efficient cost aggregation* module. Unlike existing volumetric methods, our method does not need to construct a 4D feature volume or use 3D convolutions for cost aggregation, which are the main barriers for applying volumetric methods to high-resolution images. By processing each disparity shift independently, our network only needs 2D convolutions to perform cost aggregation. We also propose a new refinement scheme using entropy, which bypasses the need to estimate

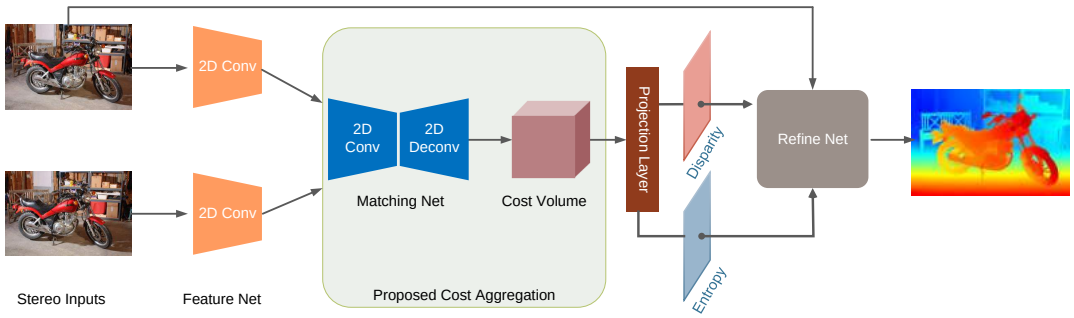


Figure 5.2: **Overall Architecture of our Stereo Network,** consisting of four components: Feature Net, Matching Net, Projection Layer and Refine Net. Given a pair of stereo images, the Feature Net produces feature maps which are processed by the Matching Net to generate a 3D cost volume (see Section 5.3.2). The Projection Layer projects the volume to obtain a disparity map while also computing an entropy map. The Refine Net takes the left image and the output of the Projection Layer to generate the final disparity map.

disparity for the right view.

5.3.1 Network Architecture

Fig. 5.2 illustrates the overall architecture of our framework, which consists of four major parts: (1) Feature Net, (2) Matching Net, (3) Projection Layer, and (4) Refine Net.

Feature Net. A deeper feature net has a larger receptive field and can extract richer information. Therefore, researchers often use more than 15 layers for feature extraction [56, 49, 50]. In practice, we find that increasing the number of layers in the feature net does not help much for the final accuracy, *e.g.*, MC-CNN [47]. Therefore, in our feature net design, we use a shallow structure that only contains 8 convolutional layers. We first downsample the input images using a 3×3 convolution with a stride of 3. Three dilated convolutions are then applied to enlarge the receptive field. We also adopt a reduced spatial pyramid pooling (SPP) module [153] to combine features from different scales to relieve the fixed-size constraint of a CNN. Our SPP module contains two average pooling layers: 64×64 and 16×16. Each of them follows a 1×1 convolution and a bilinear upsampling layer. We concatenate feature maps that feed into our SPP module, then pass them to a 3×3 convolution with output channel size of 96. The final feature map is generated by 1×1 convolution without batch normalization and activation functions. Following previous work [56, 49, 50], we use 32-channel feature maps.

Matching Net. Our Matching Net computes a cost map on each disparity level. We adopt a skip-connected U-Net [154] with some modifications. In particular, we downsample each concatenated feature map four times using 3×3 convolutions with a stride of 2. For each scale, we filter the feature maps with one 3×3 convolution followed by a batch normalization layer and a ReLU activation layer. We set the fea-

ture size of each scale at 48, 64, 96 and 128 respectively. For upsampling layers, we use 4×4 deconvolution layers with a stride of 2 and reduce the feature dimension accordingly. A 3×3 convolutional layer with feature size of 1, with no batch normalization nor activation applied to generate the final cost map. Our Matching Net has 17 layers in total.

Projection layer. After building a 3D cost volume, we use a projection layer to select the disparity with the lowest matching cost. Similar to previous volumetric methods [56, 14, 50], we use the soft-argmin operation to generate a disparity. The soft-argmin operation is defined as:

$$\hat{d} := \sum_{d=0}^{D} [d \times \sigma(-c_d)], \qquad (5.1)$$

where c_d is the matching cost at disparity d, D is the preset maximum disparity level and $\sigma(\cdot)$ denotes the softmax operator.

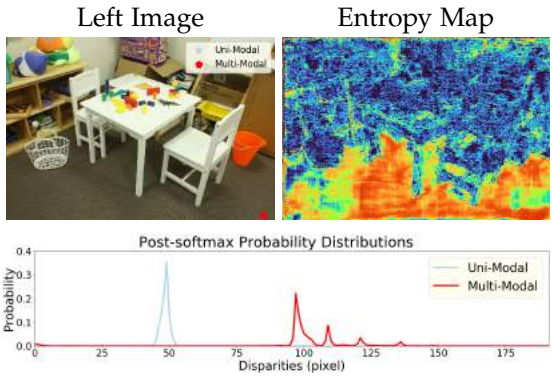


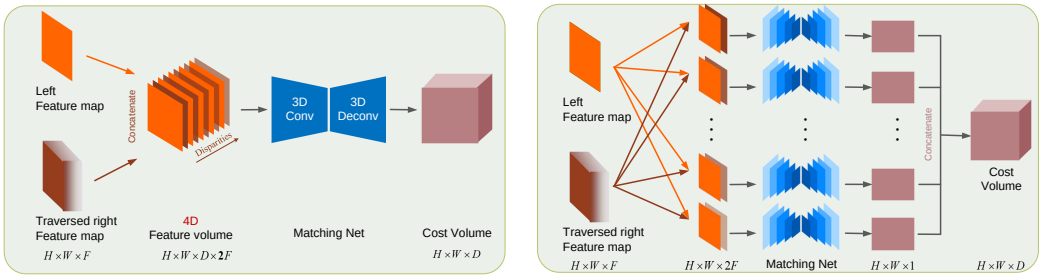
Figure 5.3: **Entropy Map Analysis.** In the entropy map, red colour corresponds to high entropy value. Best viewed in colour.

With little additional computation, the projection layer also generates an entropy map to estimate the confidence of each pixel. The entropy of each pixel is defined as:

$$h = -\sum_{d=0}^{D} \sigma(-c_d) \log(\sigma(-c_d)). \qquad (5.2)$$

We apply a softmax operation on the cost to convert it to a probability distribution before computing the entropy. Fig. 5.3 shows the resulting entropy map: pixels on texture-less areas (*e.g.*, red dot in the carpet) have high entropy, whereas the pixels on the texture-rich areas (*e.g.*, blue dot in the shelf) have low entropy. The bottom of the figure shows the post-softmax probability distribution curves of two selected pixels. A unimodal curve indicates low entropy (high confidence) for the pixel's estimated disparity, whereas a multimodal curve indicates high entropy (low confidence).

Refine Net. There are several choices in designing our refine net. StereoNet [49] proposes a refine net that takes the raw disparity map and the left image as inputs. StereoDRNet [155] uses a similar refine net but takes the disparity map, the left image, image warping error map and disparity warping error map as inputs. The main drawback of such a design is that computing these error maps requires both left and right disparity maps, which costs extra time and memory. As shown in Fig. 5.3, the entropy map can provide similar information with less computation and memory. Therefore, we use a similar refine net architecture as StereoDRNet [155] but use the disparity map, left image and entropy map as inputs.



(a) Volumetric methods (b) Proposed method

Figure 5.4: **Volumetric methods *vs.* Proposed method.** Our method runs share-weighted Matching Net on disparity shifted feature maps using 2D convolutions before 3D cost volume formation. In contrast, previous volumetric methods perform matching using 3D convolutions after constructing a 4D feature volume.

5.3.2 Efficient Cost Aggregation

Existing volumetric methods construct a 4D feature volume, which is processed with 3D convolutions to learn the matching cost, as shown in Fig. 5.4 (a). With such an approach, the matching cost for pixel $\mathbf{p}$ at disparity level d for such approaches is calculated as:

$$c_{3D}(\mathbf{p}, d) = g_{3D}(\phi_{4D}(f(I^L(\mathbf{p})) \,\|\, f(I^R(\mathbf{p} - d)))), \qquad \text{(volumetric)} \tag{5.3}$$

where $f(\cdot)$ is a feature network to convert images to feature maps, $\phi_{4D}(\cdot\|\cdot)$ denotes the concatenation of disparity-shifted feature map pairs on every possible disparity shift, and $g_{3D}(\cdot)$ is a 3D convolution-based matching network that computes and aggregates the matching cost based on feature maps and neighboring disparity shifts. Therefore, the cost will be different if we shuffle the concatenated feature maps along the disparity dimension.

In contrast, our proposed Efficient Cost Aggregation (ECA) only requires 2D convolutions. As shown in Fig. 5.4 (b), the exact same matching net is applied to each disparity-shifted feature map pair, thus ensuring that cost aggregation is only dependent on the current disparity shift. Unlike 3D convolution-based methods, our matching cost will be identical if we shuffle the concatenated feature maps along the disparity dimension.

Specifically, let us consider a matching cost computation for pixel $\mathbf{p}$ at disparity level d. In our method, we compute the matching cost as follows:

$$c_{2D}(\mathbf{p}, d) = g_{2D}(f(I^L(\mathbf{p})), f(I^R(\mathbf{p} - d))), \qquad \text{(ours)} \tag{5.4}$$

where $f(\cdot)$ is a feature net, and $g_{2D}(\cdot)$ is a matching net that computes the matching cost at each disparity level independently. Similar to MC-CNN [47], our proposed approach learns a cost from a deep network. However, MC-CNN computes the matching cost based on patches and then uses traditional semi-global cost aggre-

Table 5.1: **Computational resource comparison between Our Matching Net and Baseline3D Matching Net**. Runtime is measured on $540(H) \times 960(W)$ resolution stereo pairs with $192(D)$ disparity levels.

Methods	Runtime(s)	Params	Memory	Operations
Baseline3D Matching Net	0.150	3.84M	$\frac{1}{3}H \times \frac{1}{3}W \times \frac{1}{3}D \times 2F$	733.8 Gflops
Our Matching Net	0.001	1.16M	$\frac{1}{3}H \times \frac{1}{3}W \times 2F$	5.4 Gflops

gation steps to generate the disparity map. Our approach, on the other hand, is an end-to-end method that uses a network to perform cost computation and aggregation simultaneously.

To better understand and compare the matching cost computation, we implement *Baseline3D* using Equation (5.3) for cost computation, and we implement *Ours* using Equation (5.4) to compute the matching cost. Baseline3D and Ours networks share the same feature net and projection layer, but Baseline3D constructs a 4D feature volume as in Fig. 5.4 (a) and replaces the 2D convolutions in our Matching Net with 3D convolutions. A detailed comparison can be found in Section 5.4.3.

In Table 5.1, we compare the computational resources needed for Baseline3D and our proposed Matching Net. Note that in order to compute matching cost for each disparity level, we need to run our Matching Net $\frac{1}{3}D$ times. Since all disparity levels are processed independently, we are allowed to run our Matching Net either in parallel or sequentially. To construct a cost volume of size $\frac{1}{3}H \times \frac{1}{3}W \times \frac{1}{3}D$, the former one will need a memory of $\frac{1}{3}H \times \frac{1}{3}W \times \frac{1}{3}D \times 2F$ but can finish in $0.007s$ (including memory allocation time). The latter one just needs a memory of $\frac{1}{3}H \times \frac{1}{3}W \times 2F$ as all Matching Nets can share the same memory space and can finish in $0.07s$. We ignore the memory cost of convolution layers here as they may vary on different architectures. It is worth noting that the main obstacle for training a volumetric network on high resolution stereo pairs is that it requires the network to initialize a giant 4D feature volume in GPU memory before cost aggregation and the size of the feature volume increases as the cube of the inputs. However, since the 4D feature volume is not required for our method, we can handle high resolution inputs and large disparity ranges, *i.e.,* our method can process a pair of 1500×1000 images with 400 disparity levels in Middlebury benchmark without further downsampling the feature size to $1/64$ as in HSM [151].

Another advantage of our method is better generalization capability. Unlike volumetric methods that utilize 3D convolutions to regularize H, W, D dimensions simultaneously, we force our method to learn matching costs only from H, W dimensions. we argue that connections between $H \times W$ plane and D dimension is a double-edged sword in computing the matching costs. While such connections do improve results on a single dataset, in practice we find that they lead to higher cross-dataset errors as verified in Section 5.4.3.

5.3.3 Loss Function

Following GA-Net [50], we use the smooth ℓ_1 loss as our training loss function, which is robust at disparity discontinuities and has low sensitivity to outliers or noise. Our network outputs two disparity maps: a coarse prediction $\mathbf{d}_{\text{coarse}}$ from the soft-argmin operation and a refined one $\mathbf{d}_{\text{refine}}$ from our refine net. We apply supervision on both of them. Given the ground truth disparity $\mathbf{d}_{gt}$, the total loss function is defined as:

$$\mathcal{L}_{\text{total}} = \ell(\mathbf{d}_{\text{coarse}} - \mathbf{d}_{\text{gt}}) + \lambda\ell(\mathbf{d}_{\text{refine}} - \mathbf{d}_{\text{gt}}), \tag{5.5}$$

where $\lambda = 1.25$, and

$$\ell(x) = \begin{cases} 0.5x^2, & |x| < 1 \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{5.6}$$

5.4 Experimental Results

We implemented our stereo matching network in PyTorch. We used the same training strategy and data argumentation as described in GA-Net [50] for easy comparison. Our network was trained in an end-to-end manner with the Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We randomly cropped the input images to 240×576 and used a batch size of 104 on 8 Nvidia Tesla V100 GPUs. We pre-trained our network from random initialization and a constant learning rate of 1e-3 on the SceneFlow dataset for 60 epochs. Each epoch took around 15 minutes and the total pre-training process took around 15 hours. For inference, our network can run 100 FPS on NVIDIA Titan RTX (excluding CPU-GPU data transfer time) and occupies less than 2 GB memory for a pair of 384×1280 stereo images. We add different disparity levels as a batch and run the Matching Net over it to achieve parallelism. We comprehensively studied the characteristics of our network in ablation studies and evaluated our network on leading stereo benchmarks.

5.4.1 Datasets

Our method was evaluated on the following datasets:

SceneFlow dataset. SceneFlow [51] is a large synthetic dataset containing 35454 training and 4370 testing images with a typical image dimension of 540×960. This dataset provides both left and right disparity maps, but we only use the left ones for training. Note that for some sequences, the maximum disparity level is larger than a pre-set limit (192 for this dataset), so we exclude these pixels in our training and evaluation. We use this dataset for pre-training and ablation studies.

KITTI 2015 dataset. KITTI 2015 contains 200 real-world drive scene stereo pairs for training and 200 for testing. They have a fixed baseline but the focal lengths could vary. The typical resolution of KITTI images is 376×1240. The semi-dense ground truth disparity maps are generated by Velodyne HDL64E LiDARs with manually inserted 3D CAD models for cars [90]. From the training set, we randomly select 160 frames for training and 40 frames for validation. We use a maximum disparity level

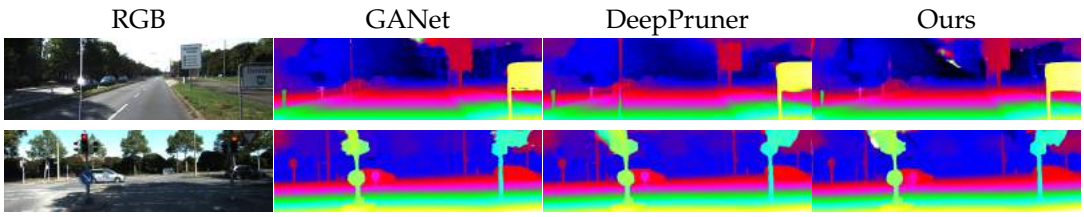


Figure 5.5: **Qualitative Results on KITTI 2015 *test* dataset.**

of 192 in this dataset. We report our results on the test set benchmark, whose ground truth is withheld.

ETH3D stereo dataset. ETH3D is a small dataset that contains 27 images for training and 20 for testing, for both indoor and outdoor scenes. It is a challenging dataset in that most stereo pairs have different lighting conditions. In other words, the Lambertian assumption for stereo matching may not hold in some areas. Moreover, unlike all the other datasets which have colour inputs, ETH3D is grayscale. It requires the network to handle different channel statistics and extract features that are robust to lighting conditions. The maximum input resolution is 576×960. Since the maximum disparity of this dataset is very small, we reduce the maximum disparity level to 48 for this dataset.

Middlebury 2014 stereo dataset. Middlebury 2014 is another small dataset. It contains 15 images for training and 15 for testing. This dataset is challenging for deep stereo methods not only because of the small size of the training set, but also due to the high resolution imagery with many thin objects. The full resolution of Middlebury is up to 3000×2000 with 800 disparity levels. To fit in GPU memory, most deep stereo methods can only operate on quarter-resolution images. As a consequence, many details are missed at that resolution which leads to a reduced accuracy. We use half resolution and 432 disparity levels.

5.4.2 Evaluation on Benchmarks

In this section, we provide results on three well-known stereo matching benchmarks: KITTI 2015, ETH3D, and Middlebury. We fine-tuned the SceneFlow pre-trained model on each benchmark. The algorithms are divided into two categories based on runtime: below 10 FPS and above 10 FPS.

KITTI 2015. Table 5.2 shows the accuracy and runtime of leading algorithms on the KITTI 2015 benchmark. Our method is the fastest and most accurate among real-time algorithms. Fig. 5.5 visualizes several results on the test set.

ETH3D. Table 5.3 shows quantitative results of our method on the ETH3D benchmark. Our method achieves competitive performance while being significantly faster than all the other methods. A qualitative comparison of our method with other SOTA algorithms is shown in Fig. 5.6.

Middlebury 2014. Our method is the only deep learning-based method that can achieve real-time performance on the Middlebury dataset. By accuracy, it is the

Table 5.2: **Results on KITTI 2015 *test* set.** Bold indicates the best, while underline indicates the second best.

			Non-occ (%)			All (%)		
	Method	Runtime(s)	bg	fg	all	bg	fg	all
Non-real time	MC-CNN [47]	67.00	2.48	7.64	3.33	2.89	8.88	3.89
	PDSnet [156]	0.50	2.09	3.68	2.36	2.29	4.05	2.58
	CRL [138]	0.47	2.32	3.68	2.36	2.48	3.59	2.67
	SDRNet [155]	0.23	1.57	4.58	2.06	1.72	4.95	2.26
	PSMnet [157]	0.41	1.71	4.31	2.14	1.86	4.62	2.32
	GC-Net [56]	0.90	2.02	3.12	2.45	2.21	6.16	2.87
	NVStereoNet [57]	0.60	2.03	4.41	2.42	2.62	5.69	3.13
	iResNet [52]	0.12	2.15	2.55	2.22	2.35	3.23	2.50
	M2S_CSPN [158]	0.50	**1.40**	**2.67**	**1.61**	**1.51**	**2.88**	**1.74**
	HSM [151]	**0.15**	1.63	3.40	1.92	1.80	3.85	2.14
	EMCUA [159]	0.90	1.50	3.88	1.90	1.66	4.27	2.09
	GA-Net-15 [50]	0.36	**1.40**	3.37	1.73	1.55	3.82	1.93
	DPruner_Best [152]	0.18	1.71	3.18	1.95	1.87	3.56	2.15
Real-time	StereoNet [49]	0.02	-	-	-	4.30	7.45	4.83
	MAD-Net [53]	0.02	3.45	8.41	4.27	3.75	9.2	4.66
	DispNetC [51]	0.04	4.11	**3.72**	4.05	4.32	**4.41**	4.34
	DeepCostAggr [160]	0.03	4.82	10.11	5.69	5.34	11.35	6.34
	RTSNet [161]	0.02	2.67	5.83	3.19	2.86	6.19	3.41
	Ours	**0.01**	**2.12**	3.88	**2.42**	2.51	4.62	**2.86**

Table 5.3: **Results on ETH3D *test* dataset**. Bold indicates the best, while underline indicates the second best.

Methods	time(s)	EPE	rmse	bad-4.0	bad-2.0	bad-1.0	A99
HSM [151]	**0.14**	0.29	0.67	0.68	1.48	4.25	3.25
SDRNet [155]	0.15	0.34	0.71	0.50	1.66	6.02	3.07
iResNet [52]	0.20	0.25	0.59	**0.34**	1.20	4.04	2.70
DPruner [152]	0.16	**0.28**	**0.58**	**0.34**	**1.04**	**3.82**	**2.61**
PSMnet [157]	0.41	0.36	0.75	0.54	1.31	5.41	3.38
DN-CSS [162]	0.07	**0.24**	**0.56**	**0.38**	**0.96**	**3.00**	2.89
Ours	**0.01**	0.32	0.63	0.53	1.25	4.82	**2.79**

Table 5.4: **Results on Middlebury 2014 *test* dataset**. Bold indicates the best, while underline indicates the second best.

Methods	time(s)	EPE	rmse	bad-4.0	bad-2.0	bad-1.0	A99
SGM [7]	0.32	5.32	20.0	12.2	18.4	31.1	109
HSM [151]	0.51	**2.07**	**10.3**	**4.83**	**10.2**	**24.6**	**39.2**
iResNet [52]	0.34	3.31	11.3	12.6	22.9	38.8	48.6
DPruner [152]	0.13	4.80	14.7	15.9	30.1	52.3	67.7
PSMNet [157]	0.64	6.68	19.4	23.5	42.1	63.9	84.5
DN-CSS [162]	0.66	4.04	13.9	14.7	22.8	36.0	58.8
Ours	**0.04**	3.12	13.8	7.22	15.4	35.1	55.6

second best among all competitors, as shown in Table 5.4. Compared to the most accurate approach (HSM), our method is 12× faster while has comparable accuracy. Qualitative results are shown in Fig. 5.7.

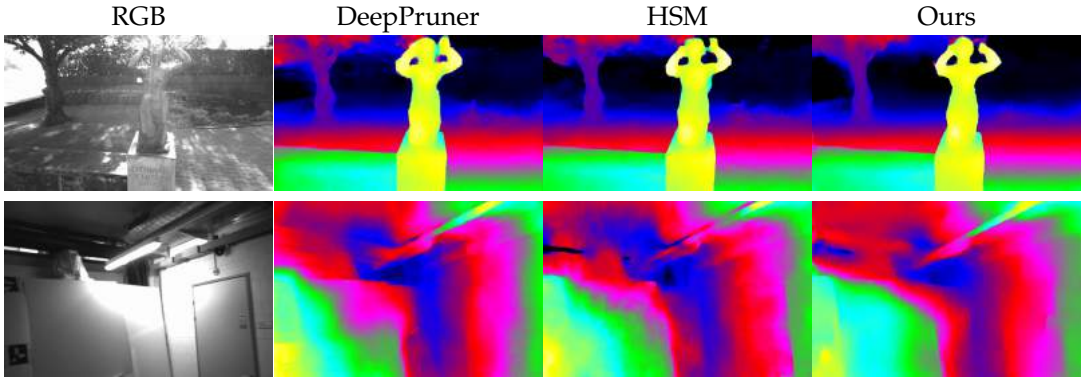


Figure 5.6: **Qualitative results on ETH3D *test* dataset.**

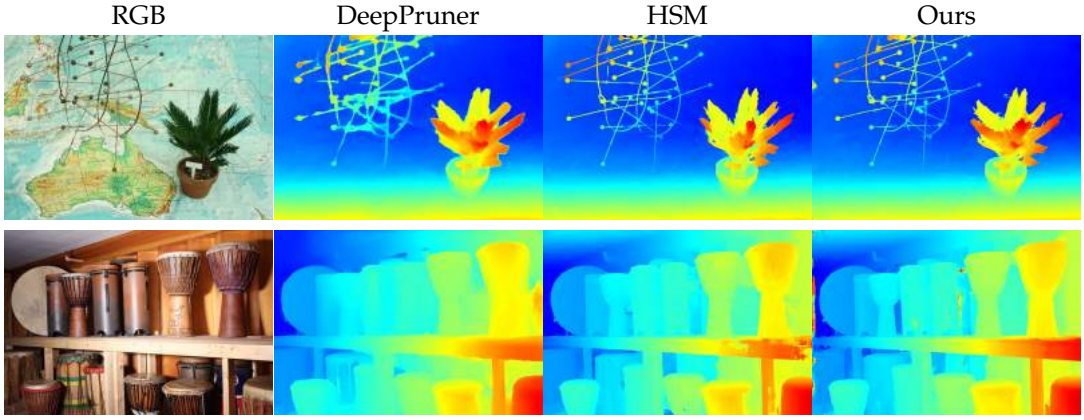


Figure 5.7: **Qualitative results on Middlebury 2014 *test* dataset.**

Table 5.5: **Contributions of each component.** The first 3 rows compare the performance (both speed and accuracy) between hand-crafted cost, Baseline3D and our method. The last 3 rows show the contribution of each components of our method.

Architecture Variant					Inference	SceneFlow	
Hand-crafted	3D	2D	RefineNet	Entropy	time(s)	EPE	bad-1.0
✓					0.001	6.05	59.13
	✓				0.15	0.78	8.11
		✓			0.007	1.20	10.31
		✓	✓		0.01	1.14	10.16
		✓	✓	✓	0.01	1.09	9.67

5.4.3 Model Design Analysis

In this section, we analyze the components of our network structure and justify the design choices, including the performance differences between our proposed method, hand-crafted cost computation and Baseline3D, the effectiveness of entropy map and the robustness and generalizability of our network. All experiments are conducted on the SceneFlow dataset except the robustness experiment.

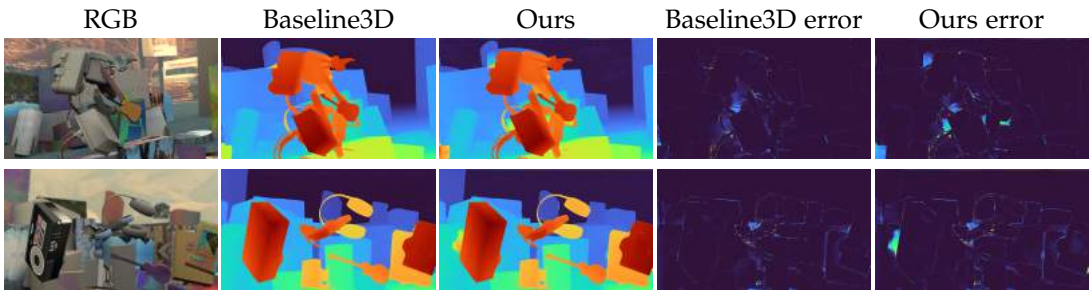


Figure 5.8: **Qualitative comparison on the SceneFlow dataset.**

5.4.3.1 Ours vs. Baseline3D.

To fairly analyze the benefits and disadvantages of our method and volumetric method (Baseline3D), we compare them using exactly the same training strategy, and settings. For the sake of completeness, we also compare the performance with hand-crafted cost using $c_{SSD}(\mathbf{p}, d) = \sum_{\mathbf{q}\in\mathcal{N}_\mathbf{p}} \left\| f(I^L(\mathbf{p})) - f(I^R(\mathbf{p}-d)) \right\|_2^2$, where $\mathcal{N}_\mathbf{p}$ is a local patch around $\mathbf{p}$.

A detailed comparison is shown in Table. 5.5. As expected, using hand-crafted costs fails to generate meaningful results. Baseline3D achieves the best performance in accuracy but is 15× slower than Ours. Compared with Baseline3D, the performance of proposed method drops from 0.78 pixels EPE to 1.09 pixels, which is acceptable in many real-world scenarios. In fact, our method achieves the top accuracy among all real-time algorithms as shown in Table. 5.6.

A qualitative comparison of Ours and Baseline3D is provided in Fig. 5.8 and Fig. 5.5, where both approaches generate high-quality disparity maps. Baseline3D has better occlusion handling ability: By jointly regularizing spatial and disparity domain, the network can better pre-scale the matching costs to handle the multi-modal situation. However, Baseline3D has a drawback that we will address in the next section. We further analyze the contribution of each component of proposed network, including the use of refine net and the entropy of the cost volume as input to the refine net. In Table. 5.5, we can see that by adding the refine net, the EPE drops from 1.20 to 1.14 and the bad-1.0 drops from 10.31 to 10.16. Adding the entropy map further boosts the accuracy more than 5%. We posit that the entropy map provides evidence for the refine net as to which parts of the disparity map are unreliable and need to be smoothed using priors.

In Table 5.6, we compare our methods with SOTA deep stereo methods on the SceneFlow dataset. Baseline3D achieves top performance (in terms of both accuracy and speed) among non-real time methods, while our method with refinement achieves the top performance among real-time methods. It is worth noting that previous 2D convolution based methods [138, 52] need a large number of parameters to learn the context mapping between inputs and disparity maps. By using proposed cost aggregation, our network achieves better accuracy, is significantly faster, and requires a fraction of the parameters as previous 2D convolution based methods.

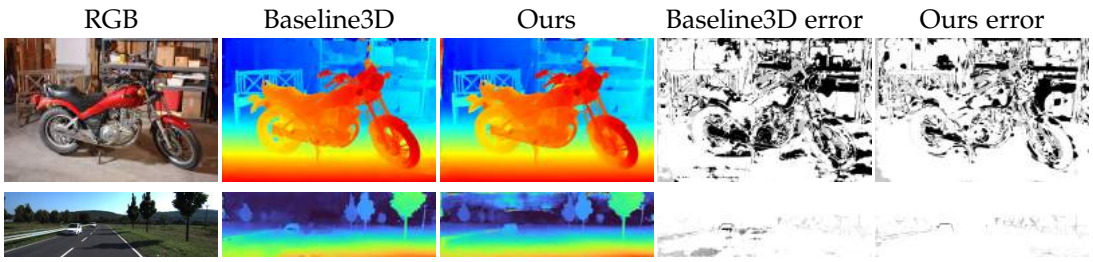


Figure 5.9: **Qualitative comparison on cross-dataset study.** The SceneFlow pre-trained models were tested on Middlebury 2014 and KITTI 2015 training sets *without any finetuning*. Note that Ours generalizes better across datasets than Baseline3D.

Table 5.6: **Quantitative results on SceneFlow dataset**. Our model is the fastest and the most accurate among real-time algorithms.

	Methods	EPE (pix)	bad-1.0	Params	Runtime(s)
Non-real time	GC-Net[56]	1.84	15.6	3.5M	0.9
	CRL[138]	1.32	-	78.77M	0.47
	iResNet[52]	2.45	9.28	43.34M	0.15
	PSMNet[157]	1.09	12.1	5.22M	0.41
	CSPN[158]	**0.78**	-	-	0.5
	SDRNet[155]	0.86	-	-	0.28
	GA-Net[50]	**0.78**	8.7	6.58M	1.9
	DPruner[152]	0.86	-	-	0.18
	Baseline3D	**0.78**	**8.11**	4.26M	**0.15**
Realtime	DispNetC[51]	1.68	-	-	0.04
	StereoNet[49]	1.10	-	-	0.02
	Ours	**1.09**	**9.67**	1.70M	**0.01**

5.4.3.2 Robustness and Generalizability

In the previous section, we showed that Baseline3D yields higher accuracy than Ours, while being significantly slower. Aside from speed and memory consumption, we consider other drawbacks in using 3D convolutions. Since spatial information should be independent of disparity, there is a risk that using 3D convolutions may cause the network to learn relationships that are present only in one specific dataset, which may actually hinder its generalization performance on other unseen datasets.

To test this hypothesis, we compared the generalizability of Ours and Baseline3D methods using the SceneFlow-pretrained models, by testing them on the Middlebury and KITTI 2015 training sets, without fine-tuning. Table 5.7 shows the quantitative results. For better comparison, we use SOTA method GA-Net [50] as the reference method. Ours consistently achieves better cross-dataset performance on these two datasets, supporting our hypothesis. The results in Fig. 5.9 show that Baseline3D generates false boundaries on the road plane while Ours does not, because the former has learned spurious associations from the training set.

Table 5.7: **Results on cross-dataset study.** SceneFlow pre-trained models were tested on Middlebury 2014 and KITTI 2015 training sets *without any finetuning*.

	Middlebury					KITTI 2015			
Methods	bad-1.0	bad-2.0	bad-4.0	avgerr	rms	EPE	bad-1.0	bad-2.0	bad-3.0
GA-Net-deep [50]	59.9	38.1	19.8	**4.94**	**13.6**	1.70	42.35%	18.29%	**10.77**%
Baseline3D	53.0	33.7	20.1	13.5	34.4	1.92	46.76%	21.02%	12.03%
Ours	**51.4**	**33.1**	**19.6**	6.67	19.2	**1.63**	**37.55**%	**17.54**%	10.88%

5.4.3.3 Random Dot Stereo

Random Dot Stereograms (RDSs) [163] were introduced many decades ago to evaluate depth perception in the human visual system. RDSs were key to establishing the fact that the human visual system is capable of estimating depth simply by matching patterns in the left and right images, without utilizing any monocular cues (lighting, colour, texture, shape, familiar objects, and so forth). Similarly, we argue that artificial stereo algorithms should be able to process random dot stereograms, and that this ability provides key evidence to understand the actual behavior of the networks, *i.e.,* context mapping or matching.

In random dot stereo pairs, there is no semantic context. Therefore, methods that fail this test, but that otherwise do well on benchmarks, are mostly likely relying on monocular, semantic cues for inference. Such methods (including DispNet, and those that build upon DispNet) will struggle to process images whose distribution varies significantly from the training distribution, because they do not actually match pixels between images. In this section, we compare our method with the recent MADNet [53] to evaluate this claim.

RDS Dataset We created a Random Dot Stereogram (RDS) dataset for the test. The dataset contains 2000 frames, 1800 for training and 200 for testing. Randomly selected chair meshes from a subset of ShapeNet Core dataset [164] are randomly positioned in scenes. Each scene contains four objects and these objects were also randomly scaled and rotated and placed at a distance between 2 to 3 meters. We added a slanted background plane to the scene at a distance between 2 and 7 meters. The world origin is randomly translated within a $0.2 \times 0.2 \times 0.2$ meter box and the z-axis of the world coordinate points towards the center of the scene. We set a virtual left camera on the left of the origin with a translation of $(0, -0.1, 0)^\top$ and a right camera on the right with a translation of $(0, 0.1, 0)^\top$. We used a fixed focal length $f = 929.426$ and a fixed image size 720×1280 and the camera center is $(c_x, c_y) = (621.749, 357.298)$. For each sample, we randomly generated a random dot texture and apply it to the meshes in the cyclopean view [165] (a virtual view placed in the middle of the left and right view). The RDS pairs were then generated by projecting the meshes to the left and right image planes with sub-pixel accuracy ground truth.

We provide qualitative and quantitative results on the RDS dataset of Ours and MADNet [53]. We fine-tuned Ours for 200 epochs with the ground truth. For MADNet, since it is a self-supervised method, we allow it to perform online fine-tuning for 200 epochs. The quantitative results are shown in Table 5.8. Unlike MADNet, Ours

Table 5.8: **Quantitative Results on RDS dataset**.

Methods	EPE	bad-1.0	bad-2.0	bad-3.0
MADNet [53]	51.21	99.42	98.84	98.26
Ours	1.02	5.45	3.59	2.93

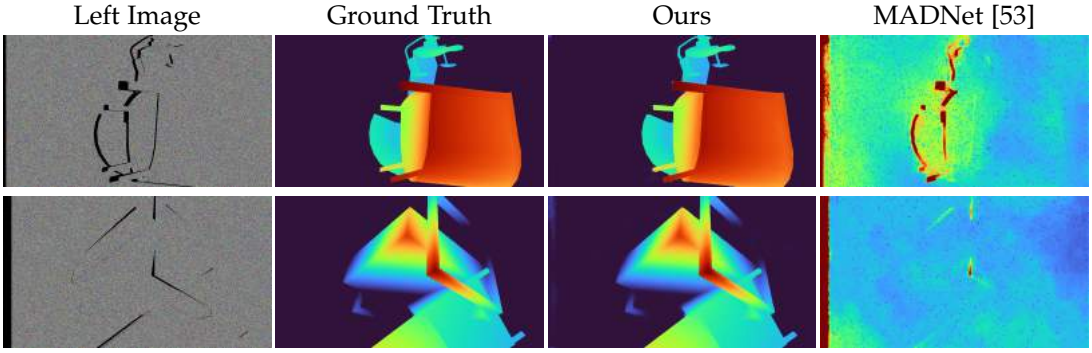


Figure 5.10: **Qualitative results on the Random Dot Stereo dataset.** Our network is able to recover disparities from RDS images while MADNet [53] can not. The black holes in the left image is due to the occlusion between the left view and the cyclopean view [165].

successfully produces high-quality disparity maps, as shown in Figure 5.10. To the best of our knowledge, our approach is the only end-to-end 2D convolution-based method that can pass the RDS test. This is because other 2D convolution-based methods are more rely on context information while our method is trying to find corresponding points with *matching*.

5.5 Conclusion

In this chapter, we have proposed a highly efficient deep stereo matching network. Our method is not only on par with the state-of-the-art deep stereo matching methods in terms of accuracy, but is also able to run in super real-time, *i.e.*, over 100 FPS on typical image sizes. This makes our method suitable for time-critical applications such as robotics and autonomous driving. The key to our success is Efficient Cost Aggregation, where 2D convolutions based cost aggregation is independently applied to all disparity levels to construct a 3D cost volume. We have extended this framework to the problem of optical flow estimation where a direct extension of volumetric methods will lead to a 5D feature volume and the requirement of 4D convolutions in [147].

Chapter 6

Hierarchical Neural Architecture Search for Deep Stereo Matching

To reduce the human efforts in neural network design, Neural Architecture Search (NAS) has been applied with remarkable success to various high-level vision tasks such as classification and semantic segmentation. The underlying idea for the NAS algorithm is straightforward, namely, to enable the network the ability to choose among a set of operations (*e.g.*, convolution with different filter sizes), one is able to find an optimal architecture that is better adapted to the problem at hand. However, so far the success of NAS has not been enjoyed by low-level geometric vision tasks such as stereo matching. This is partly due to the fact that state-of-the-art deep stereo matching networks, designed by humans, are already sheer in size. Directly applying the NAS to such massive structures is computationally prohibitive based on the currently available mainstream computing resources. In this chapter, we propose the first *end-to-end* hierarchical NAS framework for deep stereo matching by incorporating task-specific human knowledge into the neural architecture search framework. Specifically, following the gold standard pipeline for deep stereo matching (*i.e.*, , feature extraction – feature volume construction and dense matching), we optimize the architectures of the entire pipeline jointly. Extensive experiments show that our searched network outperforms all state-of-the-art deep stereo matching architectures and is ranked at the top 1 accuracy on KITTI stereo 2012, 2015 and Middlebury benchmarks, as well as the top 1 on SceneFlow dataset with a substantial improvement on the size of the network and the speed of inference[1].

6.1 Introduction

Stereo matching attempts to find dense correspondences between a pair of rectified stereo images and estimate a dense disparity map. Being a classic vision problem, stereo matching has been extensively studied for almost half a century [1]. Since MC-CNN [47], a large number of deep neural network architectures [51, 56, 52, 50] have been proposed for solving the stereo matching problem. Based on the adopted

[1]This work was originally published in [166].

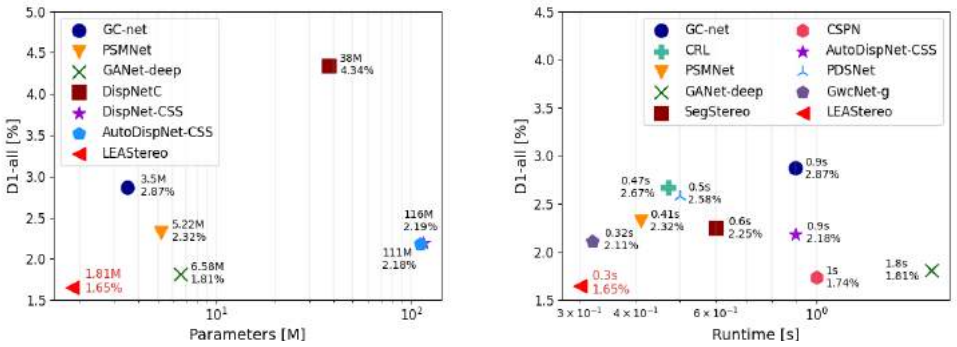


Figure 6.1: Our proposed method, LEAStereo (Learning Effective Architecture Stereo), sets a new state-of-the-art on the KITTI 2015 *test* dataset with much fewer parameters and much lower inference time.

network structures, existing deep stereo networks can be roughly classified into two categories: **I.** *direct regression* and **II.** *volumetric* methods.

Direct regression methods are based on direct regression of dense per-pixel disparity from the input images, without taking into account the geometric constraints in stereo matching [92]. In the majority of cases, this is achieved by employing large U-shape encoder-decoder networks with 2D convolutions to infer the disparity map. While enjoying a fully data-driven approach, recent studies raise some concerns about the generalization ability of the direct regression methods. For example, the DispNet [51] fails the random dot stereo tests [15].

In contrast, the volumetric methods leverage the concept of semi-global matching [7] and build a 4D feature volume by concatenating features from each disparity-shift. To this end, the volumetric methods often make use of two building blocks, the so-called **I.** *feature net* and **II.** *matching net*. As the names imply, the feature net extracts features from the input images and the matching net compute matching costs from the 4D feature volume with 3D convolutions. Different designs of the feature net and the matching net form variants of the volumetric networks [56, 14, 157, 49, 50]. Nowadays, the volumetric methods represent the state-of-the-art in deep stereo matching and top the leader-board across different benchmark datasets. Despite the success, designing a good architecture for volumetric methods remains an open question in deep stereo matching.

On a separate line of research and to reduce the human efforts in designing neural networks, Neural Architecture Search (NAS) has mounted tremendous successes in various high-level vision tasks such as classification [167, 168, 169], object detection [170, 171], and semantic segmentation [172, 173, 174]. Connecting the dots, one may assume that marrying the two parties, *i.e.*, employing NAS to design a volumetric method for stereo matching, is an easy ride. Unfortunately this is not the case. In general, NAS needs to search through a humongous set of possible architectures to pick the network components (*e.g.*, the filter size of convolution in a certain layer). This demands heavy computational load (early versions of NAS algorithms [175, 176]

need thousands of GPU hours to find an architecture on the CIFAR dataset [177]). Add to this, the nature of volumetric methods are very memory hungry. For example, the volumetric networks in [56, 157, 15, 50] require six to eight Gigabytes of GPU memory for training per batch! Therefore, end-to-end search of architectures for volumetric networks has been considered prohibitive due to the explosion of computational resource demands. This is probably why, in the only previous attempt[2], Saikia *et al.* [178] search the architecture based on the direct regression methods, and they only search partially three different cell structures rather than the full architecture.

In this chapter, we leverage the volumetric stereo matching pipeline and allow the network to automatically select the optimal structures for both the Feature Net and the Matching Net. Different from previous NAS algorithms that only have a single encoder / encoder-decoder architecture [172, 178, 174], our algorithm enables us to search over the structure of both networks, the size of the feature maps, the size of the feature volume and the size of the output disparity. Unlike AutoDispNet [178] that only searches the cell level structures, we allow the network to search for both the cell level structures and the network level structures, *e.g.*, the arrangement of the cells. To sum up, we achieve the first *end-to-end* hierarchical NAS framework for deep stereo matching by incorporating the geometric knowledge into neural architecture search. We not only avoid the explosive demands of computational resources in searching architectures, but also achieve better performances compared to naively searching an architecture in a very large search space. Our method outperforms a large set of state-of-the-art algorithms on various benchmarks (*e.g.*, topping all previous studies on the KITTI and Middlebury benchmarks[3]). This includes man-designed networks such as [50, 179] and the NAS work of Saikia *et al.* [178], not only in accuracy but also in inference time and the size of the resulting network (as shown in Figure 6.1).

6.2 Related Work

Deep Stereo Matching MC-CNN [47] is the first deep learning based stereo matching method. It replaces handcrafted features with learned features and achieves better performance. DispNet [51] is the first end-to-end deep stereo matching approach. It tries to directly regress the disparity maps from stereo pairs. The overall architecture is a large U-shape encoder-decoder network with skip connections. Since it does not leverage on pre-acquired human knowledge in stereo matching, this network is totally data-driven, and requires large training data and often hard to train. GC-Net [56] used a 4D feature volume to mimic the first step of conventional stereo matching pipeline and a soft-argmin process to mimic the second step. By encoding such human knowledge in network design, training becomes easier while maintaining high accuracies. Similar to our work, GC-Net also consists of two sub-networks to predict disparities. GA-Net [151] proposes a semi-global aggregation layer and a local

[2]To the best of our knowledge.

[3]At the time of submitting this draft, our algorithm, LEAStereo, is ranked **1** in the KITTI 2015, and KITTI 2012 benchmark, and ranked **1** according to Bad 4.0, avgerr, rms, A95, A99 metrics on Middlebury benchmark.

guided aggregation layer to capture the local and the whole-image cost dependencies respectively. Generally speaking and as alluded to earlier, designing a good structure for stereo matching is very difficult, despite considerable effort put in by the vision community.

Neural Architecture Search for Dense Predictions Rapid progress on NAS for image classification or object detection has been witnessed as of late. In contrast, only a handful of studies target the problem of dense predictions such as scene parsing, and semantic segmentation. Pioneer works [180, 181] propose a super-net that embed a large number of architectures in a grid arrangement and adopt them for the task of semantic segmentation. To deal with the explosion of computational demands in dense prediction, Chen *et al.* [182] employ a handcrafted backbone and only search the decoder architecture. Rather than directly searching architectures on large-scale dense prediction tasks, they design a small scale proxy task to evaluate the searching results. Nekrasov *et al.* [183] focus on the compactness of a network and utilized a small-scale network backbone with over-parameterised auxiliary searchable cells on top of it. Similarly, Zhang *et al.* [173] penalized the computational demands in search operations, allowing the network to search an optimized architecture with customized constraints. Auto-Deeplab [172] proposes a hierarchical search space for semantic segmentation, allowing the network to self-select spatial resolution for encoders. FasterSeg [174] leverages on the idea of [173] and [172], and introduces multi-resolution branches to the search space to identify an effective semantic segmentation network. AutoDispNet [178] applies NAS to disparity estimation by searching cell structures for a large-scale U-shape encoder-decoder structure.

6.3 Our Method

In this section, we present our end-to-end hierarchical NAS stereo matching network. In particular, we have benefited from decades of human knowledge in stereo matching and previous successful handcrafted designs in the form of priors towards architecture search and design. By leveraging task-specific human knowledge in the search space design, we not only avoid the explosion demands of computational resources in searching architectures for high resolution dense prediction tasks, but also achieve better accuracy compared to naively searching an architecture in a very large search space.

6.3.1 Task-specific Architecture Search Space

We recall that the deep solutions for dense prediction (*e.g.*, semantic segmentation, stereo matching), usually opt for an encoder-decoder structure [172, 178, 174]. Inspired by the Auto-DeepLab [172] for semantic segmentation, we propose a two-level hierarchical search that allows us to identify both cell-level and network-level structures[4]. Directly extending ideas from semantic segmentation might not necessarily

[4]We would like to stress that our framework, in contrast to the Auto-DeepLab, searches for the *full* architecture (Auto-DeepLab only searches for the architecture of the encoder).

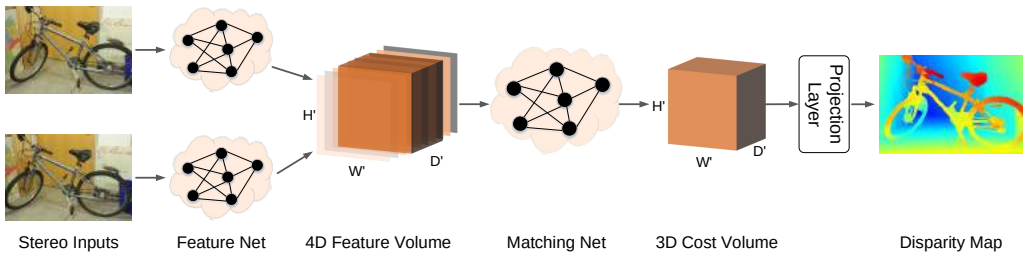


Figure 6.2: **The pipeline of our proposed stereo matching network.** Our network consists of four components: 2D Feature Net, 4D Feature Volume, 3D Matching Net, and Projection Layer. Given a pair of stereo images, the Feature Net produces feature maps that are processed by the Matching Net to generate a 3D cost volume. The disparity map can be projected from the cost volume with soft-argmin operation. Since the Feature Net and the Matching Net are the only two modules that contain trainable parameters, we utilize the NAS technique to select the optimal structures for them.

lead to viable solutions for stereo matching. A fully data-driven U-shape encoder-decoder network is often hard to train, even with the help of NAS [178] in regressing disparity maps. Volumetric stereo matching methods offer faster convergence and better performance as their pipeline makes use of inductive bias (*i.e.*, human knowledge in network design). To be specific, volumetric solutions first obtain a matching cost for all possible disparity levels at every pixel (based on the concepts of 3D geometry) and then use it to generate the disparity map (*e.g.*, by using a soft-argmin operation). One obvious drawback here is the overwhelming size of the resulting network. This, makes it extremely difficult, if not impossible, to use volumetric solutions along the NAS framework.

In this work, we embed the geometric knowledge for stereo matching into our network architecture search. Our network consists of four major parts: a 2D feature net that extracts local image features, a 4D feature volume, a 3D matching net to compute and aggregate matching costs from concatenated features, and a soft-argmin layer that projects the computed cost volumes to disparity maps. Since only the feature net and the matching net involve trainable parameters, we leverage NAS technique to search these two sub-networks. The overall structure of our network is illustrated in Figure 6.2. More details about the cell and network level search are presented below.

6.3.1.1 Cell Level Search Space

A cell is defined as a core searchable unit in NAS. Following [184], we define a cell as a fully-connected directed acyclic graph (DAG) with $\mathcal{N}$ nodes. Our cell contains two input nodes, one output node and three intermediate nodes. For a layer l, the output node is $\mathcal{C}_l$ and the input nodes are the output node of its two preceding layers (*i.e.*, $\mathcal{C}_{l-2}, \mathcal{C}_{l-1}$). Let $\mathcal{O}$ be a set of candidate operations (*e.g.*, 2D convolution, skip connection). During the architecture search, the functionality of an intermediate

node $s^{(j)}$ is described by:

$$\boldsymbol{s}^{(j)} = \sum_{i \rightsquigarrow j} o^{(i,j)}\left(\boldsymbol{s}^{(i)}\right) . \tag{6.1}$$

Here, $\rightsquigarrow$ denotes that node i is connected to j and

$$o^{(i,j)}(\boldsymbol{x}) = \sum_{r=1}^{\nu} \frac{\exp\left(\alpha_r^{(i,j)}\right)}{\sum_{s=1}^{\nu} \exp\left(\alpha_s^{(i,j)}\right)} o_r^{(i,j)}(\boldsymbol{x}) , \tag{6.2}$$

with $o_r^{(i,j)}$ being the r-th operation defined between the two nodes. In effect, to identify the operation connecting node i to node j, we make use of a mixing weight vector $\boldsymbol{\alpha}^{(i,j)} = (\alpha_1^{(i,j)}, \alpha_2^{(i,j)}, \cdots, \alpha_\nu^{(i,j)})$ along a softmax function. At the end of the search phase, a discrete architecture is picked by choosing the most likely operation between the nodes. That is, $o^{(i,j)} = o_{r^*}^{(i,j)}$; $r^* = \arg\max_r \alpha_r^{(i,j)}$. Unlike [184, 178], we only need to search one type of cells for the feature and matching networks since the change of spatial resolution is handled by our network level search. DARTS [184] has a somehow inflexible search mechanism, in the sense that nodes $\mathcal{C}_{l-2}, \mathcal{C}_{l-1}, \mathcal{C}_l$ are required to have the same spatial and channel dimensionalities. We instead allow the network to select different resolutions for each cell. To handle the divergence of resolutions in neighbouring cells, we first check their resolutions and adjust them accordingly by upsampling or downsampling if there is a mismatch.

Residual Cell. Previous studies opt for concatenating the outputs of all intermediate nodes to form the output of a cell (*e.g.*, [184, 172, 178]). We refer to such a design as a direct cell. Inspired by the residual connection in ResNet [185], we propose to also include the input of the cell in forming the output. See Figure 6.3 where the residual connection cells highlighted with a red line. This allows the network to learn *residual* mappings on top of direct mappings. Hereafter, we call this design *the residual cell*. We empirically find that residual cells outperform the direct ones (see § 6.4.3).

Candidate Operation Selection. The candidate operations for the feature net and matching net differ due to their functionalities. In particular, the feature net aims to extract distinctive local features for comparing pixel-wise similarities. We empirically observe that removing two commonly used operations in DARTS, namely **1.** *dilated separable convolutions* and **2.** *the pooling layers* does not hurt the performance. Thus, the set of candidates operators for the feature net includes $\mathcal{O}^F = \{$ "3×3 2D convolution", "skip connection"$\}$. Similarly, we find out that removing some of the commonly used operations from the candidate set for the matching net will not hurt the design. As such, we only include the following operations for the matching net, $\mathcal{O}^M = \{$ "$3 \times 3 \times 3$ 3D convolution", "skip connection"$\}$. We will shortly see an ablation study regarding this (see § 6.4.3).

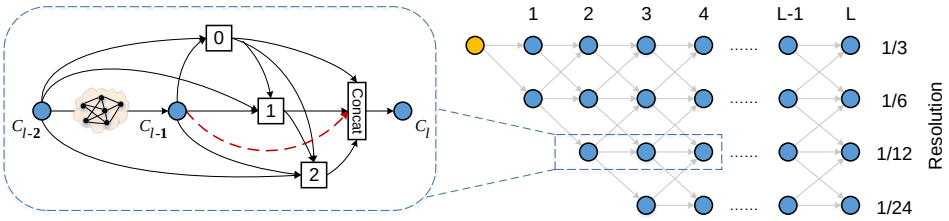


Figure 6.3: **The proposed search space.** We illustrate our cell level search space on the left and our network level search space on the right. The red dash line on the left represents the proposed residual connection. We set $L^F = 6$ for Feature Net and $L^M = 12$ for Matching Net.

6.3.1.2 Network Level Search Space

We define the network level search space as the arrangement of cells, which controls the variations in the feature dimensionality and information flow between cells. Drawing inspirations from [172], we aim to find an optimal path within a pre-defined L-layer trellis as shown in Figure 6.3. We associated a scalar with each gray arrow in that trellis. We use β to represent the set of this scalar. Considering the number of filters in each cell, we follow the common practice of doubling the number when halving the height and width of the feature tensor.

In the network level search space, we have two hyper-parameters to set: **I.** the smallest spatial resolution and **II.** the number of layers L. Empirically, we observe that setting the smallest spatial resolution to $1/24$ of the input images work across a broad range of benchmarks and hence our choice in here. Based on this, we propose a four-level trellis with downsampling rates of $\{3, 2, 2, 2\}$, leading to the smallest feature map to be $1/24$ of the input size (see Figure 6.3). In comparison to $\{2, 2, 2, 2, 2\}$, the downsampling to $1/3$ will remove the need of upsampling twice and we empirically observed similar performance. At the beginning of the feature net, we have a three-layer "stem" structure, the first layer of it is a 3×3 convolution layer with stride of three, followed by two layers of 3×3 convolution with stride of one.

Turning our attention to the the number of layers L, we have empirically observed that choosing $L^F = 6$ for the feature net and $L^M = 12$ for the matching net provides a good balance between the computational load and performance of the network. This is interestingly much smaller than some recent developments in hand-crafting deep stereo matching networks. For example, GA-Net [50] uses 33 convolutional layers with an hourglass structure to extract features.

Similar to finding the best operation between the nodes, we will use a set of search parameters $\boldsymbol{\beta}$ to search over the trellis in order to find a path in it that minimizes the loss. As shown in Fig 6.3, each cell in a level of the trellis can receive inputs from the preceding cell in the same level, one level below and one level above (if any of the latter two exists).

6.3.2 Loss Function and Optimization

Since our network can be searched and trained in an end-to-end manner, we directly apply supervisions on the output disparity maps, allowing the Feature Net and the Matching Net to be jointly searched. We use the smooth ℓ_1 loss as our training loss function as it is considered to be robust to disparity discontinuities and outliers. Given the ground truth disparity $\mathbf{d}_{gt}$, the loss function is defined as:

$$\mathcal{L} = \ell(\mathbf{d}_{\text{pred}} - \mathbf{d}_{\text{gt}}), \text{ where } \ell(x) = \begin{cases} 0.5x^2, & |x| < 1, \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{6.3}$$

After continuous relaxation, we can optimize the weight $\mathbf{w}$ of the network and the architecture parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ through bi-level optimization. We parameterize the cell structure and the network structure with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively. To speed up the search process, we use the first-order approximation [172].

To avoid overfitting, we use two disjoint training sets train**I** and train**II** for $\mathbf{w}$ and $\boldsymbol{\alpha}, \boldsymbol{\beta}$ optimization respectively. We do alternating optimization for $\mathbf{w}$ and $\boldsymbol{\alpha}, \boldsymbol{\beta}$:

- Update network weights $\mathbf{w}$ by $\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ on train**I**
- Update architecture parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ by $\nabla_{\boldsymbol{\alpha},\boldsymbol{\beta}} \mathcal{L}(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ on train**II**

When the optimization convergence, we decode the discrete cell structures by retaining the top-2 strongest operations from all non-zero operations for each node, and the discrete network structures by finding a path with maximum probability [172].

6.4 Experiments

In this section, we adopt SceneFlow dataset [51] as the source dataset to analyze our architecture search outcome. We then conduct the architecture evaluation on KITTI 2012 [89], KITTI 2015 [90] and Middlebury 2014 [149] benchmarks by inheriting the searched architecture from SceneFlow dataset. In our ablation study, we analyze the effect of changing search space as well as different search strategies.

6.4.1 Architecture Search

We conduct full architecture search on SceneFlow dataset [51]. It contains 35,454 training and 4370 testing synthetic images with a typical image dimension of 540×960. We use the "finalpass" version as it is more realistic. We randomly select 20,000 image pairs from the training set as our search-training-set, and another 1,000 image pairs from the training set are used as the search-validation-set following [172].

Implementation: We implement our LEAStereo network in Pytorch. Random crop with size 192×384 is the only data argumentation technique being used in this work. We search the architecture for a total of 10 epochs: the first three epochs to initiate the weight $\mathbf{w}$ of the super-network and avoid bad local minima outcome; the rest epochs to update the architecture parameters α, β. We use SGD optimizer with momentum 0.9, cosine learning rate that decays from 0.025 to 0.001, and weight

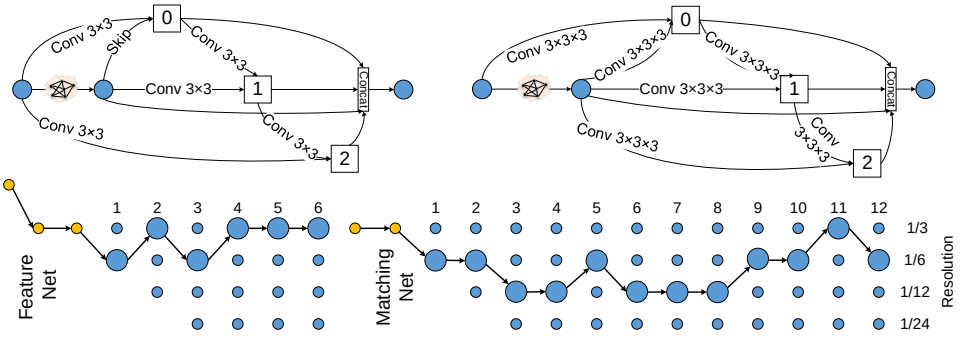


Figure 6.4: **The searched architecture.** The top two graphs are the searched cell structures for the Feature Net and Matching Net. The bottom are the searched network level structures for both networks. The yellow dots present the pre-defined "stem" layers and the blue dots denote the searchable cells.

decay 0.0003. The entire architecture search optimization takes about 10 GPU days on an NVIDIA V100 GPU.

The architecture found by our algorithm is shown in Figure 6.4. We manually add 2 skip connections for the Matching Net, one is between node 2 and 5, the other is 5 and 9. Noting that we only perform the architecture search once on the SceneFlow dataset and fine-tuned the searched architecture weights on each benchmark separately.

6.4.2 Benchmark Evaluation

SceneFlow dataset We evaluate our LEAStereo network on SceneFlow [51] test set with 192 disparity level. We use average end point errors (EPE) and bad pixel ratio with 1 pixel threshold (bad 1.0) as our benchmark metrics. In Table 6.1, we can observe that LEAStereo achieves the best performance using only near one third of parameters in comparison to the SOTA hand-crafted methods (*e.g.,* [50]). Furthermore, the previous SOTA NAS-based method AutoDispNet [178] has 20× more parameters than our architecture. We show some of the qualitative results in Figure 6.7.

Table 6.1: **Quantitative results on Scene Flow dataset**. Our method achieves state-of-the-art performance with only a fraction of parameters. The parentheses indicate the test set is used for hyperparameters tuning.

Methods	**Params** [M]	**EPE** [px]	**bad 1.0** [%]	**Runtime** [s]
GCNet [56]	3.5	1.84	15.6	0.9
iResNet[52]	43.34	2.45	9.28	0.2
PSMNet [157]	5.22	1.09	12.1	0.4
GANet-deep [50]	6.58	0.78	8.7	1.9
AANet [179]	3.9	0.87	9.3	**0.07**
AutoDispNet [178]	37	(1.51)	-	0.34
LEAStereo	**1.81**	**0.78**	**7.82**	0.3

KITTI benchmarks As shown in Table 6.2 and the leader board, our LEAStereo achieves top 1 rank among other human designed architectures on KITTI 2012 and KITTI 2015 benchmarks. Figure 6.5 provides some visualizations from the testsets.

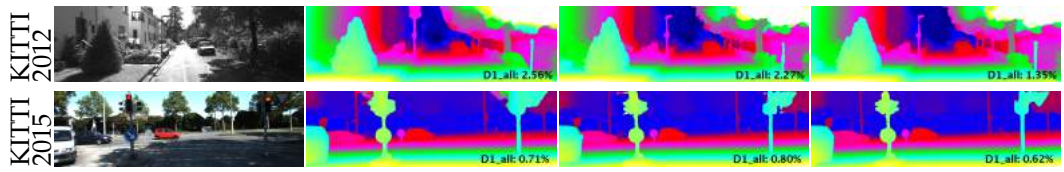


Figure 6.5: **Qualitative results on KITTI 2012 and KITTI 2015 benchmarks.**

Table 6.2: **Quantitative results on the KITTI 2012 and 2015 benchmark.** Bold indicates the best.

	KITTI 2012				KITTI 2015			
Methods	**bad 2.0** noc [%]	**bad 3.0** noc [%]	**Refl 3.0** noc [%]	**Avg All** [px]	**FG** Non Occ[%]	**Avg All** Non Occ[%]	**FG** All Areas[%]	**Avg All** All Areas[%]
GCNet [56]	2.71	1.77	10.80	0.7	5.58	2.61	6.16	2.87
PSMNet [157]	2.44	1.49	8.36	0.6	4.31	2.14	4.62	2.32
GANet-deep [50]	**1.89**	1.19	6.22	0.5	3.39	1.84	3.91	2.03
DispNetC [51]	7.38	4.11	16.04	1.0	3.72	4.05	4.41	4.34
AANet [179]	2.90	1.91	10.51	0.6	1.80	2.32	1.99	2.55
AutoDispNet-CSS [178]	2.54	1.70	5.69	0.5	2.98	2.00	3.37	2.18
LEAStereo	1.90	**1.13**	**5.35**	**0.5**	**2.65**	**1.51**	**2.91**	**1.65**

Middlebury 2014 Middlebury 2014 is often considered to be the most challenging dataset for deep stereo networks due to restricted number of training samples and also the high resolution imagery with many thin objects. The full resolution of Middlebury is up to 3000×2000 with 800 disparity levels which is prohibitive for most deep stereo methods. This has forced several SOTA [157, 152] to operate on quarter-resolution images where details can be lost. In contrast, the compactness of our searched architecture compactness allows us to use images of size 1500×1000 with 432 disparity levels. In Table 6.3, we report the latest benchmark of volumetric based stereo matching method [157] and direct regression approaches [52, 179] for comparisons. Our proposed LEAStereo achieves the state-of-the-art rank on various metrics (*e.g.*, bad 4.0, all) among more than 120 stereo methods from the leader board. We also note that our network has better performance on large error thresholds, which might because of the downsampling operations prohibit the network to learn sub-pixel accuracy or the loss function that does not encourage sub-pixel accuracy.

Table 6.3: **Quantitative results on the Middlebury 2014 Benchmark.** Bold indicates the best. The red number on the top right of each number indicates the actual ranking on the benchmark.

Methods	bad 2.0 [%]		bad 4.0 [%]		Ave Err [px]		RMSE [px]		A95 [px]	
	nonocc	all	nonocc	all	nonocc	all	nonocc	all	nonocc	all
PSMNet [157]	42.1[108]	47.2[104]	23.5[97]	29.2[94]	6.68[69]	8.78[43]	19.4[50]	23.3[22]	31.3[65]	43.4[32]
iResNet [52]	22.9[62]	29.5[61]	12.6[55]	18.5[49]	3.31[30]	4.67[7]	11.3[8]	13.9[6]	12.5[37]	20.7[7]
AANet+ [179]	15.4[44]	22.0[40]	10.8[45]	16.4[42]	6.37[66]	9.77[49]	23.5[73]	29.4[42]	48.8[80]	76.1[63]
HSM [151]	10.2[26]	16.5[19]	4.83[17]	9.68[8]	2.07[2]	3.44[2]	10.3[4]	13.4[3]	4.32[6]	17.6[4]
LEAStereo	**7.15**[10]	**12.1**[5]	**2.75**[1]	**6.33**[1]	**1.43**[1]	**2.89**[1]	**8.11**[1]	**13.7**[5]	**2.65**[1]	**6.35**[1]

Left image HSM [151] HSM [151] bad 4.0 LEAStereo LEAStereo bad 4.0

Figure 6.6: **Qualitative results on Middlebury 2014 Benchmark.**

6.4.3 Ablation Study

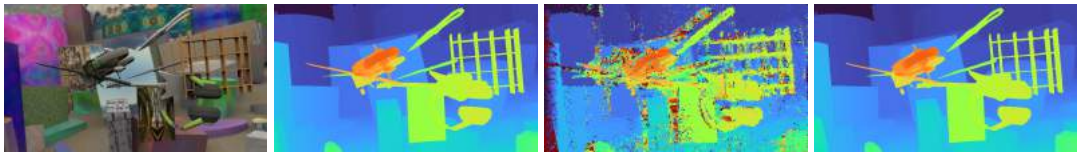
In this part, we perform ablation studies using the SceneFlow dataset [51] to justify "hyper-parameters" of our algorithm. In particular, we look into the candidate operations $\mathcal{O}$, differences between the residual and the direct cells, joint-search *vs.* separate-search of the Feature Net and the Matching Net, and functionality analysis for each sub-net. We also provide a head-to-head comparison to AutoDispNet [178].

Table 6.4: **Ablation Studies of different searching strategies.** The input resolution is 576 × 960, and EPE is measured on total SceneFlow test set.

Architecture Variant						Achieved Network		
$\mathcal{O}_{large}$	$\mathcal{O}_{ours}$	Sepa.	Join.	Direct	Residual	Params	FLOPS	EPE
√		√			√	0.68M	682.8G	1.55px
	√	√			√	2.00M	782.4G	0.86px
	√		√	√		1.52M	538.9G	0.91px
	√		√		√	1.81M	782.2G	0.78px

Candidate Operations Here we evaluate two sets of candidate operations $\mathcal{O}_{large}$ and $\mathcal{O}_{ours}$. The $\mathcal{O}_{large}$ consists of 8 operations including various types of convolution filters, pooling and connection mechanisms that are commonly used in [184, 172, 178][5]. The $\mathcal{O}_{ours}$ as described in § 6.3.1.1 only contains 3 × 3 convolution, skip connection and zero connection. For the Matching Net, we simply use the 3D variants of

[5]Namely, $\mathcal{O}_{large}$ contains 3 × 3 and 5 × 5 depth-wise separable convolutions, 3 × 3 and 3 × 3 dilated separable convolutions with dilation factor 2, skip connection, 3 × 3 average pooling, 3 × 3 max pooling and zero connection.



Left image GT Feature Net result Full Network result

Figure 6.7: **Functionality analysis for the Feature Net and the Matching Net.** By using the learned features from the Feature Net directly, we already achieve reasonably good results as shown in the third sub-figure. After applying the Matching Net, we could further improve the results significantly as evidenced by the gap between the third and fourth sub-figures.

these operations. As shown in the first two rows of Table 6.4, larger set of operations pool $\mathcal{O}_{large}$ leads to a searched architecture with a much lower number of parameters but poor EPE performance when compared to $\mathcal{O}_{ours}$. The reason is that the algorithm favors skip connections in its design, leading to a low-capacity architecture. Similar observations, albeit in another context, are reported in [186].

Joint-search *vs.* Separate-search To analyze the effectiveness and efficiency of joint-search vs. separate-search, we report their performances on SceneFlow dataset with $\mathcal{O}$ and connection type fixed. From the metrics of row 2 and 4 in Table 6.4, we observe that joint-search outperforms separate-search by an improved margin of 9.30% for EPE and 10.50% reduction on the number of parameters. We conjecture that the joint-search increases the capacity and compatibility for both Feature Net and Matching Net.

Residual cell *vs.* Direct cell We then study the differences between our proposed residual cell and direct cell. As shown in the last two rows of Table 6.4, using residual cell slightly generates more parameters and FLOPs but increase the performance by 14.29%.

Functionality analysis for each sub-net The Feature Net and the Matching Net own different roles in stereo matching. The Feature Net is designed to extract distinctive features from stereo pairs while the Matching Net is to compute matching costs from these features. To analyze and reflect the actual behaviour of each searched sub-net, we use the features from the Feature Net to directly compute a cost volume with dot products [47] and project it to a disparity map with the Winner-Takes-All (WTA) strategy. As shown in Figure 6.7, this strategy already achieves a reasonably good result in correctly estimating disparity for most objects, which demonstrates that our Feature Net is learning discriminative features for stereo matching. The difference between the third and fourth sub-figures (before and after the Matching Net) proves the contribution of the Matching Net in computing and aggregating the matching costs to achieve much better results.

6.4.4 Discussion

LEAStereo *vs.* AutoDispNet AutoDispNet [178] has a very different network design philosophy than ours. It is a large U-Net-like architecture and tries to directly regress disparity maps from input images in pixel space. In contrast, our design benefits from task-specific physics and inductive bias, *i.e.*, the gold standard pipeline

for deep stereo matching and the refine search space, thus achieves full architecture search within current physical constraints. Table 6.5 provides a head-to-head comparison on KITTI Benchmark. It is worth noting that our method is 32.12% better than AutoDispNet-CSS [178] with only 1.81M parameters (which is 1.7% of the parameters required by AutoDispNet-CSS!)

Table 6.5: **LEAStereo *vs.* AutoDispNet**

Methods	Search Level	Params	KITTI 2012	KITTI 2015	Runtime
AutoDispNet-CSS	Cell	111M	1.70%	2.18%	0.9 s
LEAStereo	Full Network	1.8M	1.13%	1.65%	0.3 s

Hints from the found architecture There are several hints from the found architecture: 1. The feature net does not need to be too deep to achieve a good performance; 2. Larger feature volumes lead to better performance (1/3 is better than 1/6); 3. A cost volume of 1/6 resolution seems proper for good performance; 4. Multi-scale fusion seems important for computing matching costs (*i.e.,* using a DAG to fuse multi-scale information). Note that the fourth hint has also been exploited by AANet [179] which builds multiple multi-scale cost volumes and regularizes them with a multi-scale fusion scheme. Our design is tailored for the task of stereo matching and may not be considered as a domain agnostic solution for a different problem. For different domains, better task-dependent NAS mechanisms are still a suitable solution to efficiently incorporate inductive bias and physics of the problem into the search space.

6.5 Conclusion

In this chapter, we proposed the first end-to-end hierarchical NAS framework for deep stereo matching, which incorporates task-specific human knowledge into the architecture search framework. Our search framework tailors the search space and follows the feature net-feature volume-matching net pipeline, whereas we could optimize the architecture of the entire pipeline jointly. Our searched network outperforms all state-of-the-art deep stereo matching architectures (handcrafted and NAS searched) and is ranked at the top 1 accuracy on KITTI stereo 2012, 2015 and Middlebury benchmarks while showing substantial improvement on the network size and inference speed. In the future, we plan to extend our search framework to other dense matching tasks such as optical flow estimation [147] and multi-view stereo [187].

Chapter 7

Noise-Aware Unsupervised Deep Lidar-Stereo Fusion

In this charpter, we present LidarStereoNet, the first unsupervised Lidar-stereo fusion network, which can be trained in an end-to-end manner without the need of ground truth depth maps. By introducing a novel "Feedback Loop" to connect the network input with output, LidarStereoNet could tackle both noisy Lidar points and misalignment between sensors that have been ignored in existing Lidar-stereo fusion work. Besides, we propose to incorporate the piecewise planar model into the network learning to further constrain depths to conform to the underlying 3D geometry. Extensive quantitative and qualitative evaluations on both real and synthetic datasets demonstrate the superiority of our method, which outperforms state-of-the-art stereo matching, depth completion and Lidar-Stereo fusion approaches significantly [1].

7.1 Introduction

Perceiving the surrounding 3D information from passive and active sensors accurately is crucial for numerous applications such as localization and mapping [188], autonomous driving [189], obstacle detection and avoidance [190], and 3D reconstruction [191, 45]. However, each kind of sensors alone suffers from its inherent drawbacks. Stereo cameras are well-known for suffering from computational complexities and their incompetence in dealing with textureless/repetitive areas and occlusion regions [192], while Lidar sensors can often provide accurate but relatively sparse depth measurements [193].

Therefore, it is highly desired to fuse measurements from Lidar and stereo cameras to achieve high-precision depth perception by exploiting their complementary properties. However, it is a non-trivial task as accurate Stereo-Lidar fusion requires a proper registration between Lidar and stereo images. Existing methods are not fully satisfactory due to the following drawbacks:

- Existing deep neural network based Lidar-Stereo fusion work [194, 195, 196] strongly depend on the availability of large-scale ground truth depth maps,

[1] This work was originally published in [3]

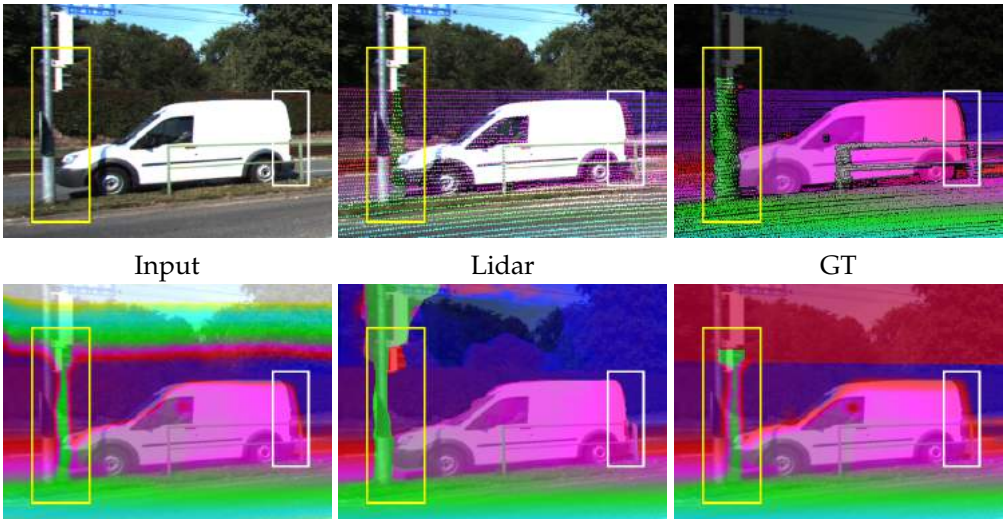


Figure 7.1: **Results on KITTI 2015.** We highlight the displacement error of Lidar points with bounding boxes. Lidar points are dilated for better visualization and we overlay our disparity maps to the colour images for illustration. Note the Lidar points for the foreground car and utility pole have been aligned to the background. Our method successfully recovers accurate disparities on tiny and moving objects while the other methods are misled by drifted and noisy Lidar points.

thus their performance are fundamentally limited by the generalization ability to real-world applications.

- Due to the rolling-shutter effect of Lidar and other calibration imperfections, a direct registration will introduce obvious alignment errors between Lidar depth and stereo depth. Furthermore, existing methods tend to assume the Lidar measurements are noise-free [124, 35]. However, as illustrated in Fig. 7.1, the misalignment and noisy Lidar measurements cause significant defects in stereo-Lidar fusion.

In this chapter, we tackle the above challenges and propose a novel framework "LidarStereoNet" for accurate Stereo-Lidar fusion, which can be trained in an end-to-end unsupervised learning manner. Our framework is noise-aware in the sense that it explicitly handles misalignment and noise in Lidar measurements.

Firstly, we propose to exploit the photometric consistency between stereo images, and the depth consistency between stereo cameras and Lidar to build an unsupervised training loss, thus removing the need of ground truth depth/disparity maps. The benefit of our novel network design is twofold: **1**) this enables a strong generalization ability of our framework in various real-world applications; **2**) our network design allows the sparsity of input Lidar points to be varied, and could even handle an extreme case when the Lidar sensor is completely unavailable.

Secondly, to alleviate noisy Lidar measurements and slight misalignment between stereo and Lidar, we present a novel training strategy that gradually remove these

noisy points during the training process automatically. Furthermore, we have also presented a novel structural loss (plane fitting loss) to handle the inaccurate Lidar measurements and stereo matching.

Under our problem setting, we make no assumption on the inputs such as the pattern/number of Lidar measurements, the probability distribution of Lidar points or stereo disparities. The end-to-end unsupervised learning nature enables good generalization ability of our network.

Experimental results on different datasets demonstrate that our method is able to recover highly accurate depth maps through Lidar-Stereo fusion that outperforms existing stereo matching methods, depth completion methods and Lidar stereo fusion methods with a large margin (at least twice better than previous ones). To the best of our knowledge, there is no deep learning based method available yet that can achieve this goal under our problem setting.

7.2 Related Work

Stereo Matching. Deep convolutional neural networks (CNNs) based stereo matching methods have recently achieved great success. Existing supervised deep methods either formulate the task as depth regression [51] or multi-label class classifications [47]. Recently, unsupervised deep stereo matching methods have also been introduced to remove the requirement of large amount of labeled training data. Godard *et al.* [109] proposed to exploit the photometric consistency loss between left image and the warped version of right image, thus achieving unsupervised stereo matching. Zhong *et al.* [15] presented a stereo matching network for estimating depths from continuous video input. Very recently, Zhang *et al.* [197] extended the self-supervised stereo network [14] from passive stereo cameras to active stereo scenarios. Even though stereo matching has been greatly advanced, it still suffers from challenging scenarios such as texture-less and low-lighting conditions.

Depth Completion/ Interpolation. Lidar scanners can provide accurate but sparse and incomplete 3D measurements. Therefore, there is a highly desired requirement in increasing the density of Lidar scans, which is crucial for applications such as self-driving cars. Uhrig *et al.* [35] proposed a masked sparse convolution layer to handle sparse and irregular Lidar inputs. Chodosh *et al.* [198] utilized compressed sensing to approach the sparsity problem for scene depth completion. With the guidance of corresponding colour images, Ma *et al.* [124] extended the up-projection blocks proposed by [125] as decoding layers to achieve full depth reconstruction. Jaritz *et al.* [39] presented a method to handle sparse inputs of various densities without any additional mask input.

Lidar-Stereo Fusion. Existing works maily focus on fusing stereo and time-of-flight (TOF) cameras for indoor scenarios [199, 200, 201, 202, 203], whereas Lidar and stereo cameras fusion for outdoor scenes has been seldom approached in the literature. Badino *et al.* [194] used Lidar measurements to reduce the searching space for stereo matching and provided predefined paths for dynamic programming. Later on, Mad-

dern *et al.* [195] proposed a probabilistic model to fuse Lidar and disparities by combining prior from each sensor. However, their performance drops significantly when the Lidar information is missing. To tackle this issue, instead of using a manually selected probabilistic model, Park *et al.* [196] utilized CNNs to learn such a model, which takes two disparities as input: one from the interpolated Lidar and the other from semi-global matching [7]. Compared with those supervised approaches, our unsupervised method can be trained in an end-to-end manner using stereo pairs and sparse Lidar points without using external stereo matching algorithms.

7.3 Lidar-Stereo Fusion

In this section, we formulate Lidar-stereo fusion as an unsupervised learning problem and present our main ideas in dealing with the inherent challenges faced by existing methods.

7.3.1 Problem Statement

Lidar-stereo fusion aims at recovering a dense and accurate depth/disparity map from sparse Lidar measurements $S \in \mathbb{R}^{n\times 3}$ and a pair of stereo images I_l, I_r. We assume the Lidar and the stereo camera have been calibrated with extrinsic matrix T and the stereo camera itself is calibrated with intrinsic matrices K_l, K_r and projection matrices P_l, P_r. We can then project sparse Lidar points S onto the image plane of I_l by $d_l^s = P_l T S$. Since disparity is used in the stereo matching task, we convert the projected depth points d_l^s to disparity D_l^s using $D = Bf/d$, where B is the baseline between the stereo camera pair and f is the focal length. The same process is applied to the right images as well. Mathematically this problem can be defined as:

$$(\widehat{D}_l, \widehat{D}_r) = \mathcal{F}(I_l, I_r, D_l^s, D_r^s; \theta), \tag{7.1}$$

where $\mathcal{F}$ is the learned Lidar-stereo fusion model (a deep network in our paper) parameterized by θ.

Under our problem setup, we do not make any assumption about the Lidar points' configuration (*e.g.,* the number or the pattern) or error distributions of Lidar points and stereo disparities. Removing all these restrictions makes our method more generic, hence having wider applicability.

7.3.2 Dealing with Noise

In Lidar-stereo fusion, existing methods usually assume the Lidar points are noise free and the alignment between the Lidar and stereo images are perfect. We would like to argue that even for dedicated system such as the KITTI dataset, the Lidar points are never perfect and the alignment could also be inaccurate, *cf.* Fig. 7.1. It is known that the ability of deep CNNs to overfit or memorize the corrupted labels

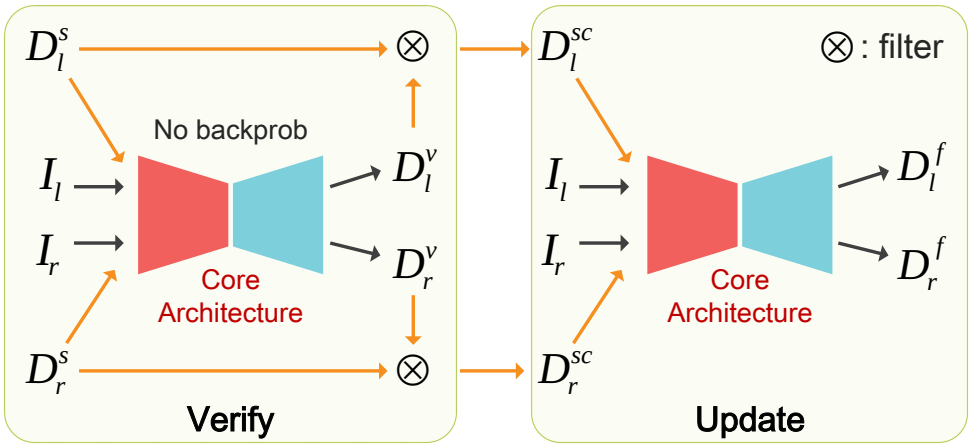


Figure 7.2: **The feedback loop.** For each iteration, the input stereo pair first computes initial disparities to filter errors in sparse Lidar points. At this stage, no *backprob* is taken place. So we call it the Verify phase. Then in the Update phase, the Core Architecture takes stereo pairs and cleaned sparse Lidar points as inputs to generate the final disparities. The parameters of the Core Architecture will be updated through *backprob* this time.

can lead to very poor generalization performance. Therefore, we aim to deal with the noise in Lidar measurements properly to train deep CNNs.

Robust functions such the ℓ_1 norm, Huber function or the truncated ℓ_2 norm are natural choices in dealing with noisy measurements. However, these functions will not eliminate the effects caused by noises but only suppress them. Further, these errors also exist in the input, so letting the network automatically ignore these error points creates an extra difficulty. To this end, we introduce a *feedback loop* in our network design to allow the input also to depend on the output of the network. In this way, the input Lidar points can be cleaned before feeding into the network.

The Feedback Loop We propose a novel framework to gradually detect and remove erroneous Lidar points during the training process and generate a highly accurate Lidar-Stereo fusion. Figure 7.2 illustrates an unfolded structure of our network design, namely the feedback loop, which consists of two phases: "Verify" phase and "Update" phase. Each phase shares the same network structure of Core Architecture, details of which will be explained in Section 7.4.1.

In the Verify phase, the network takes stereo image pairs (I_l, I_r) and noisy Lidar disparities (D_l^s, D_r^s) as input, and generates two disparity maps (D_l^v, D_r^v). No back-propagation takes place in this phrase. We then compare (D_l^v, D_r^v) and (D_l^s, D_r^s) and keep the remaining sparse Lidar points (D_l^{sc}, D_r^{sc}) that are consistent in both stereo matching and Lidar measurements. In the Update phase, the network takes both stereo pairs (I_l, I_r) and cleaned sparse Lidar points (D_l^{sc}, D_r^{sc}) as the inputs to recover dense disparity maps (D_l^f, D_r^f). All loss functions are evaluated on the final disparity outputs (D_l^f, D_r^f) only. Once the network has been trained, there will be no need for

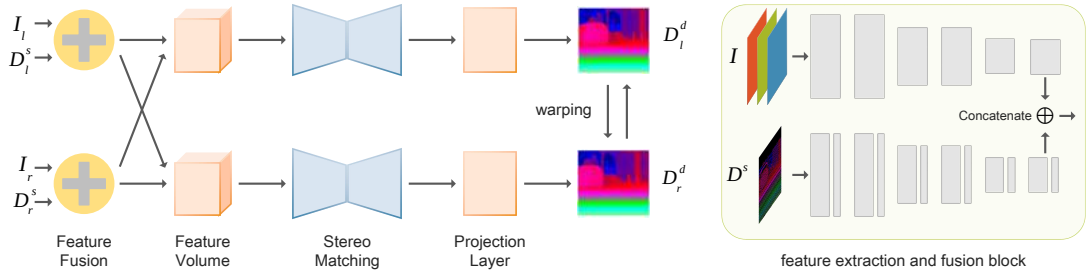


Figure 7.3: **Core Architecture of our LidarStereoNet.** It consists of a feature extraction and fusion block, a stack-hourglass type feature matching block and a disparity computing layer. Given a stereo pair I_l, I_r and corresponding projected Lidar points D_l^s, D_r^s, the feature extraction block produces feature maps separately for images and Lidar points. The feature maps are then concatenated to form final input features which are aggregated to form a feature volume. The feature matching block learns the cost of feature-volume. Then we use the disparity computing layer to obtain disparity estimation. Details of the feature extraction and fusion block is illustrated on the right.

the feedback loop. Therefore, we can use the Core Architecture only in testing to reduce the processing time.

7.4 Our Network Design

In this section, we present our new "LidarStereoNet" for Lidar-Stereo fusion, which can be learned in an unsupervised end-to-end manner. To avoid the need of large-scale training data with ground truth, we propose to exploit the photometric consistency between stereo images, and the depth consistency between stereo cameras and Lidar. This novel network design enables the following benefits: **1**) A wide generalization ability of our framework in various real-world scenarios; **2**) Our network design allows the sparsity of input Lidar points to be varied, and could even handle the extreme case when the Lidar sensor is completely unavailable. Furthermore, to alleviate the noise in Lidar measurements and the misalignment between Lidar and stereo cameras, we introduce the "Feedback Loop" into the network design to connect the output to the input, which enables the Lidar points being cleaned before feeding into the network.

7.4.1 Core Architecture

We illustrate the detailed structure of the Core architecture of our LidarStereoNet in Fig. 7.3. LidarStereoNet consists of the following blocks: 1) Feature extraction and fusion; 2) Feature matching and 3) Disparity computing. The general information flow of our network is similar to [15] but has some crucial modifications in the feature extraction and fusion block. In view of different characteristics between dense colour

images and sparse disparity maps, we leverage different convolution layers to extract features from each of them. For colour images, we use the same feature extraction block from [157] while for sparse Lidar inputs, sparse invariant convolution layer [35] is used to extract features. The final feature maps are produced by concatenating stereo image features and Lidar features. Feature maps from left and right branches are concatenated to form a 4D feature volume with a maximum disparity range of 192. Then feature matching is processed through a hourglass structure of 3D convolutions to compute matching cost at each disparity level. Similar to [56], we use the soft-argmin operation to produce a 2D disparity map from the cost volume.

Dealing with dense and sparse inputs. To extract features from sparse Lidar points, Uhrig *et al.* [35] proposed a sparsity invariant CNN after observing the failure of conventional convolutions. However, Ma *et al.* [124] and Jaritz *et al.* [39] argued that using a standard CNN with special training strategies can achieve better performance and can also handle varying input densities. We compared both approaches and realized that standard CNNs can handle sparse inputs and even get better performance but they request much deeper network (ResNet38 encoded VS 5 Convolutional layers) with 500 times more trainable parameters (13675.25K VS 25.87K). Using such a "deep" network as a feature extractor will make our network not feasible for end-to-end training and hard to converge. Therefore, in our network, we choose sparsity invariant convolutional layers to assemble our Lidar feature extractor. It consists of 5 sparse convolutional layers with a stride of 1. Each convolution has an output channel of 16 and is followed by a ReLU activation function. We attached a plain convolution with a stride of 4 to generate the final 16 channels Lidar features in order to make sure the Lidar features compatible with the image features.

7.4.2 Loss Function

Our loss function consists of two data terms and two regularization terms. For data terms, we directly choose the image warping error $\mathcal{L}_w$ as a dense supervision for every pixel and discrepancy on filtered sparse Lidar points $\mathcal{L}_l$. For the regularization terms, we use colour weighted smoothness term $\mathcal{L}_p$ and our novel slanted plane fitting loss $\mathcal{L}_p$. Our overall loss function is a weighted sum of the above loss terms:

$$\mathcal{L} = \mathcal{L}_l + \mu_1 \mathcal{L}_w + \mu_2 \mathcal{L}_s + \mu_3 \mathcal{L}_p, \tag{7.2}$$

where $\mu_1 = 1, \mu_2 = 0.001, \mu_3 = 0.01$.

7.4.2.1 Image Warping Loss

We assume photometric consistency between stereo pairs such that corresponding points between each pair should have similar appearance. However, in some cases, this assumption does not hold, so we also compare the difference between small patches' Census transform as it is robust for photometric changes. Our image warp-

ing loss is thus defined as follow:

$$\mathcal{L}_w = \mathcal{L}_i + \lambda_1 \mathcal{L}_c + \lambda_2 \mathcal{L}_g, \tag{7.3}$$

where $\mathcal{L}_i$ stands for photometric loss, $\mathcal{L}_c$ represents Census loss and $\mathcal{L}_g$ is the image gradient loss. We set $\lambda_1 = 0.1, \lambda_2 = 1$ for the balance of each term.

The photometric loss is defined as the difference between the observed left (right) image and the warped left (right) image, where we have weighted each term with the observed pixels to account for the occlusion.

$$\mathcal{L}_i = \left[\sum_{i,j} \varphi \left(\hat{I}(i,j) - I(i,j) \right) \cdot O(i,j) \right] / \sum_{i,j} O(i,j), \tag{7.4}$$

where $\varphi(s) = \sqrt{s^2 + 0.001^2}$ and the occlusion mask O is computed through left-right consistency check.

To further improve the robustness in evaluating the image warping error, we used the Census transformation to measure the difference:

$$\mathcal{L}_c = \left[\sum_{i,j} \varphi \left(\hat{C}(i,j) - C(i,j) \right) \cdot O(i,j) \right] / \sum_{i,j} O(i,j). \tag{7.5}$$

Lastly, we have also used the difference between image gradients as an error metric:

$$\mathcal{L}_g = \left[\sum_{i,j} \varphi \left(\nabla \hat{I}(i,j) - \nabla I(i,j) \right) \cdot O(i,j) \right] / \sum_{i,j} O(i,j). \tag{7.6}$$

7.4.2.2 Lidar Loss

The cleaned sparse Lidar points after our feedback verification can also be used as a sparse supervision for generating disparities. We leverage the truncated ℓ_2 function to handle noises and errors in these sparse Lidar measurements,

$$\mathcal{L}_l = ||M(\widehat{D} - D^{sc})||_\tau, \tag{7.7}$$

where M is the mask computed in the Verify phase. The truncated ℓ_2 is defined as:

$$||\cdot||_\tau = \begin{cases} 0.5x^2, & |x| < \tau \\ 0.5\tau^2, & \text{otherwise.} \end{cases} \tag{7.8}$$

7.4.2.3 Smoothness Loss

The smoothness term in the loss function is defined as:

$$\mathcal{L}_s = \sum \left(e^{-\alpha_1 |\nabla I|} \, |\nabla d| + e^{-\alpha_2 \left| \nabla^2 I \right|} \left| \nabla^2 d \right| \right) / N, \tag{7.9}$$

where $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$. Note that previous works [109] [14] often neglect the weights α_1, α_2, which actually play a crucial role in colour weighted smoothness term.

7.4.2.4 Plane Fitting Loss

We also introduce a slanted plane model into deep learning framework to enforce structural constraint. This model has been commonly used in conventional Conditional Random Field (CRF) based stereo matching/optical flow algorithms. It assumes that all pixels within a superpixel lie on a 3D plane.

By leveraging this piecewise plane fitting loss, we could enforce strong regularization on 3D structure. Although our slanted plane model is defined on disparity space, it has been proved that a plane in disparity space is still a plane in 3D space [204]. Mathematically, the disparity d_p of each pixel p is parameterized by a local plane,

$$d_p = a_p u + b_p v + c_p, \tag{7.10}$$

where (u, v) is the image coordinate, the triplet (a_p, b_p, c_p) denotes the parameters of a local disparity plane.

Define P as the matrix representation of pixel's homogeneous coordinates within a superpixel with a size of $N \times 3$ where N is number of pixels within a segment, and denote $\mathbf{a}$ as the planar parameters. Given the current disparity predictions $\widehat{\mathbf{d}}$, we can estimate the plane parameter in closed-form via $\mathbf{a}^* = (P^T P)^{-1} P^T \widehat{\mathbf{d}}$. With the estimated plane parameter, the fitted planer disparities $\widetilde{\mathbf{d}} \in \mathbb{R}^N$ can be computed as $\widetilde{\mathbf{d}} = P\mathbf{a}^* = P(P^T P)^{-1} P^T \widehat{\mathbf{d}}$.

Our plane fitting loss then can be defined as

$$\mathcal{L}_p = \|\widehat{\mathbf{d}} - \widetilde{\mathbf{d}}\| = \|[I - P(P^T P)^{-1} P^T]\widehat{\mathbf{d}}\|. \tag{7.11}$$

7.5 Experiments

We implemented our LidarStereoNet in Pytorch. All input images were randomly cropped to 256×512 during training phases while we used their original size in inference. The typical processing time of our net was about 0.5 fps on Titan XP. We used the Adam optimizer with a constant learning rate of 0.001 and a batch size of 1. We performed a series of experiments to evaluate our LidarStereoNet (LSN) on both real-world and synthetic datasets. In addition to analyzing the accuracy of depth predictions in comparison to previous work, we also conducted a series of ablation studies on different sensor fusing architectures and investigate how each component of the proposed loss contribute to the performance.

7.5.1 KITTI Dataset

The KITTI dataset [89] is created to set a benchmark for autonomous driving visual systems. It captures depth information from a Velodyne HDL-64E Lidar and corresponding stereo images from a moving platform. They use a highly accurate inertial measurement unit to accumulate 20 frames of raw Lidar depth data in a reference frame and serves as ground truth for the stereo matching benchmark. In KITTI 2015 [205], they also take moving objects into consideration. The dynamic objects are first

Table 7.1: **Quantitative results on the selected KITTI 141 subset.** We compare our LidarStereoNet with various state-of-the-art Lidar-Stereo fusion methods, where our proposed method outperforms all the competing methods with a wide margin. PSMnet-ft is fine-tuned on KITTI VO dataset. Note that, even without Lidar measurements available, our result still outperforms these methods.

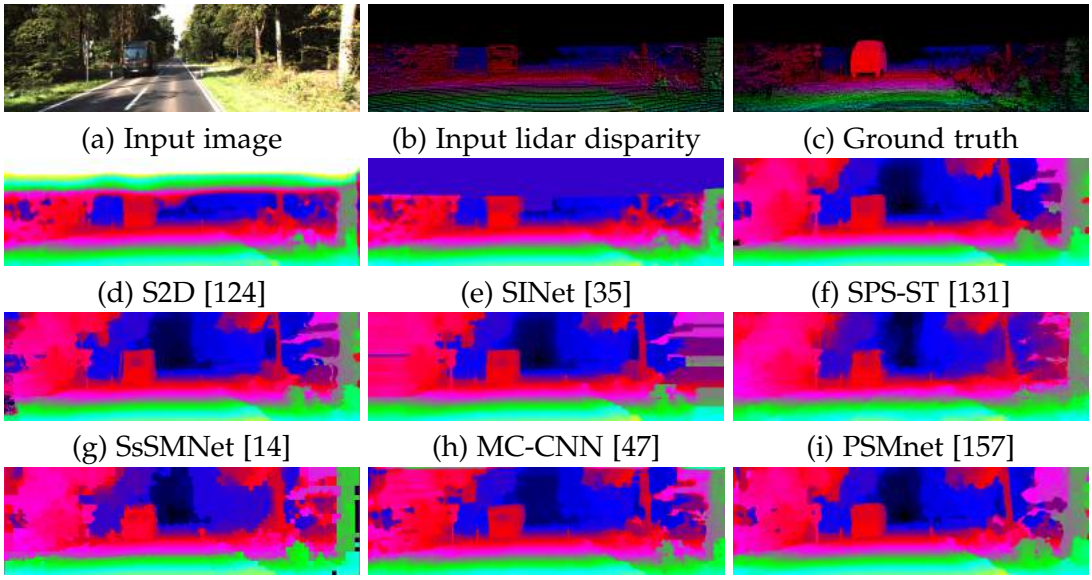
Methods	Input	Supervised	Abs Rel	> 2 px	> 3 px	> 5 px	$\delta < 1.25$	Density
MC-CNN [47]	Stereo	Yes	0.0798	0.1070	0.0809	0.0555	0.9472	100.00%
PSMnet-ft [157]	Stereo	Yes	0.0609	0.0635	0.0410	0.0277	0.9689	100.00%
SPS-ST [131]	Stereo	No	0.0633	0.0702	0.0413	0.0265	0.9660	100.00%
SsSMnet [14]	Stereo	No	0.0619	0.0743	0.0498	0.0334	0.9633	100.00%
Input Lidar	Lidar	-	-	0.0572	0.0457	0.0375	-	7.27%
S2D [124]	Lidar	Yes	0.0665	0.0849	0.0659	0.0430	0.9626	100.00%
SINet [35]	Lidar	Yes	0.0659	0.0908	0.0660	0.0456	0.9576	100.00%
Prob fusion [195]	Stereo + Lidar	No	-	-	0.0591	-	-	99.6%
CNN Fusion [196]	Stereo + Lidar	Yes	-	-	0.0484	-	-	99.8%
Our method	Stereo	No	**0.0572**	**0.0540**	**0.0345**	**0.0220**	**0.9731**	100.00%
Our method	Stereo + Lidar	No	**0.0350**	**0.0287**	**0.0198**	**0.0126**	**0.9872**	100.00%

removed and then re-inserted by fitting CAD models to the point clouds, resulting in a clean and dense ground truth for depth evaluation.

Dataset Preparation. After these processes, the raw Lidar points and the ground truth differ extremely in terms of outliers and density as shown in Fig. 7.1. In raw data, due to the large displacement between the Lidar and the stereo cameras [204], boundaries of objects may not perfectly align when projecting Lidar points onto image planes. Also, since Lidar system scans depth in a line by line order, it will create a "rolling shutter" effect on the reference image, especially on a moving platform. Instead of heuristically removing measurements, our method is able to omit these outliers automatically which is evidently shown in Fig. 7.1.

We use the KITTI VO dataset [89] as our training set. We sorted all 22 KITTI VO sequences and found 7 frames from sequence 17 and 20 that have corresponding frames in KITTI 2015 training set. Therefore we excluded these two sequences and used the remaining 20 stereo sequences as our training dataset. Our training dataset contains 42104 images with a typical image size of 1241×376. To achieve sparse disparities as inputs, we project raw Lidar points onto left and right images using provided extrinsic and intrinsic parameters and convert the raw Lidar depths to disparities. Maddern *et al.* [195] also traced 141 frames from KITTI raw dataset that have corresponding frames in the KITTI 2015 dataset and reported their results on this subset. To fulfil the consistency of evaluation, we used the same subset to evaluate our performance and utilize the 6 frames from KITTI VO dataset as our validation set (there is 1 frame that overlaps the KITTI 141 subset, so we excluded it from our validation).

Comparisons with State-of-the-Art. There are two sub-tasks that involved in our problem setting. One is stereo matching and the other one is depth completion. Here, we compare our approach with state-of-the-art methods in each sub-task.



(a) Input image (b) Input lidar disparity (c) Ground truth

(d) S2D [124] (e) SINet [35] (f) SPS-ST [131]

(g) SsSMNet [14] (h) MC-CNN [47] (i) PSMnet [157]

(j) Probabilistic fusion [195] (k) CNN fusion [196] (l) Ours

Figure 7.4: **Qualitative results of the methods from Table 7.1.** Our method is trained on KITTI VO dataset and tested on the selected unseen KITTI 141 subset without any finetuning.

For stereo matching, we use SPS-ST [131], MC-CNN [47], PSMnet [157] and SsSMnet [14]. For depth completion, we compare with S2D [124] and SINet [35]. Only the SPS-ST method is a traditional (non-deep) method, and its meta-parameters were tuned on KITTI dataset. For deep MC-CNN we used a model which was firstly trained on Middlebury dataset and for PSMnet we used the model that was trained on SceneFlow [51] dataset for the sake of fair comparison. We also compared our method with state-of-the-art self-supervised stereo matching network SsSMnet [14]. In our implementation of S2D and SINet, we trained them on KITTI depth completion dataset [35]. From 151 training sequences, we excluded 28 sequences that overlaps with KITTI 141 dataset and used the remaining 123 sequences to train these networks from scratch in a supervised manner. As a reference, we also computed the input Lidar's error rate. It is worth noting that our method increases the disparity density from less than 7.3% to 100% while reducing the error rate by a half.

We also compared our method with two existing Lidar-stereo fusion methods: Probabilistic fusion [195] and CNN fusion [196] and outperform them with a large margin.

Quantitative comparison between our method and the competing state-of-the-art methods is reported in Table 7.1, from which one can clearly see that our self-supervised LidarStereoNet achieves the best performance throughout all the metrics evaluated. Note that, our method even outperforms recent supervised CNN based fusion method [196] with a large margin. More qualitative evaluations of our method in challenging scenes are provided in Fig. 7.4. These results demonstrate the supe-

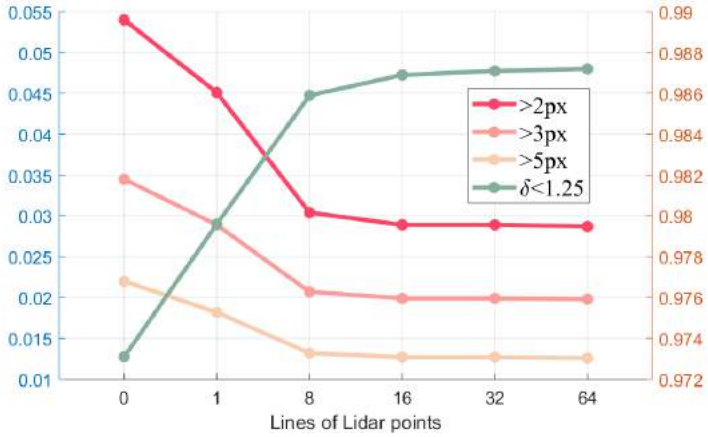


Figure 7.5: **Test results of our network on the selected KITTI 141 subset with varying levels of input Lidar points sparsity.** Left column: lower is better; right column: higher is better.

riority of our method that can effectively leverage the complementary information between Lidar and stereo cameras.

On Input Sparsity. Thanks to the nature of deep network and sparsity invariant convolution, our LidarStereoNet can handle Lidar input of varying density, ranging from no Lidar input to 64 lines input. To see this trend, we downsampled the vertical and horizontal resolution of the Lidar points. As shown in Fig. 7.5, our method performs equally well when using 8 or more lines of Lidar points. Note that even when we are using zero Lidar points as input (in this case, the problem become a pure stereo matching problem), we still outperform SOTA stereo matching methods.

Table 7.2: **Ablation study on the feedback loop** Type 1 and Type 2 show the performance only use the Core Architecture without and with removing error lidar points from the input, while Full model means our proposed feedback loop.

Methods	Abs Rel	> 2 px	> 3 px	> 5 px
Type 1	0.0539	0.0411	0.0310	0.0229
Type 2	0.0468	0.0401	0.0302	0.0226
Full model	**0.0350**	**0.0287**	**0.0198**	**0.0126**

7.5.2 Ablation Study

In this section, we perform ablation studies to evaluate the importance of the feedback loop and proposed losses in our LidarStereoNet. Notably, all ablation studies on losses and fusion strategies are evaluated on Core Architecture only in order to reduce the randomness introduced by our feedback loop module.

Importance of the feedback loop. We evaluate the importance of our proposed feed-

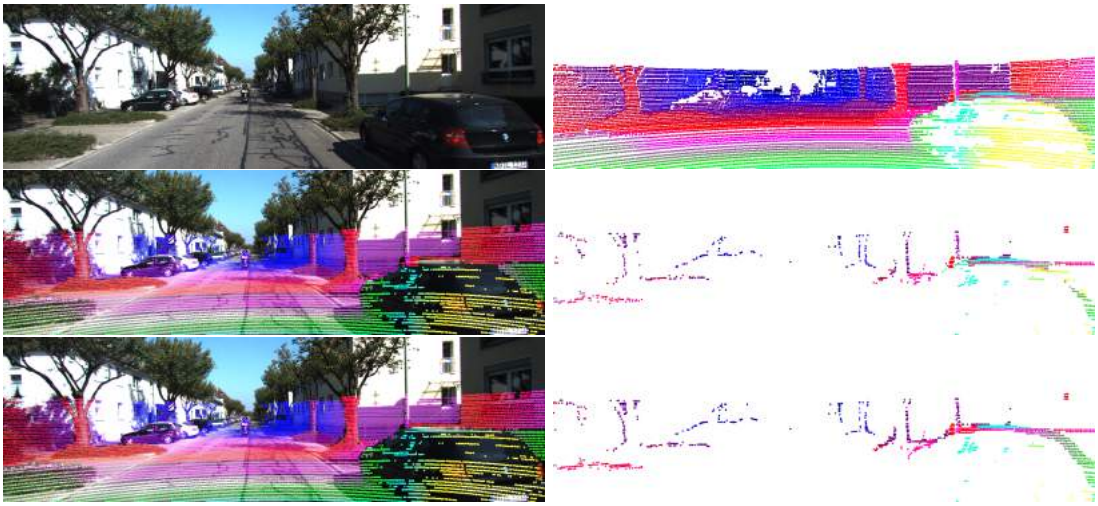


Figure 7.6: **Gradually cleaned input Lidar points.** From top to bottom, left column: left image, cleaned Lidar points at the 2^{nd} epoch, cleaned Lidar points at the 5^{th} epoch; Right column: raw Lidar points, error points find at the 2^{nd} epoch, error points find at the 5^{nd} epoch. Note that the error measurements on the right car has been gradually removed.

back loop in two aspects. One is by removing this module and only keep the Core Architecture, we call it *Type 1*. The other is also use the Core Architecture remove these inconsistent points in computing losses, we use *Type 2* to represent. Note that for both types, we use raw lidar points as input.

As shown in Table 7.2, adding the feedback loop significantly reduces the error rate from 3.01% to 1.99%. And it also shows that only removing error points from the loss computation part is not enough. These errors need to be excluded from the input as well.

Comparing different loss functions. Unlike CRF based methods, we do not require a recovered disparity map to be strictly constructed by a set of slanted planes. Instead, we use it as a soft constraint that can be balanced by other constraints such as smoothness term and warping term.

Table 7.3 shows the performance-gain with different losses. As we can see, when only using Lidar points as supervision, its performance is affected by these outliers in Lidar measurements. Adding a warping loss can reduce the error rate from 4.71% to 3.02%.

Comparing different fusion strategies. Considering the problem of utilizing sparse depth information, one naive approach will be directly using Lidar measurements for supervisions. As shown in Table 7.4, its performance is affected by the misaligned Lidar points and it has a relatively high error rate of 4.10%. The second method to leverage the depth as a fourth channel additionally to the RGB image. We call it an early fusion strategy. As shown in Table 7.4, it has the worst performance among the baselines. This may be due to the incompatible characteristics between RGB images and depth maps such that the network is unable to handle well within a common

Table 7.3: **Evaluation of different loss functions.** $\mathcal{L}_w$, $\mathcal{L}_s$, $\mathcal{L}_l$ and $\mathcal{L}_p$ represent warping loss, smoothness loss, Lidar loss and plane fitting loss separately.

Loss	Abs Rel	$>$ 2 px	$>$ 3 px	$>$ 5 px
$\mathcal{L}_l$	0.0555	0.0733	0.0471	0.0296
$\mathcal{L}_w + \mathcal{L}_s$	0.0628	0.0940	0.0637	0.0405
$\mathcal{L}_w + \mathcal{L}_s + \mathcal{L}_l$	0.0565	0.0401	0.0302	0.0226
$\mathcal{L}_w + \mathcal{L}_s + \mathcal{L}_l + \mathcal{L}_p$	**0.0468**	**0.0393**	**0.0276**	**0.0201**

Table 7.4: **Comparison of different fusion strategies.**

Methods	Abs Rel	$>$ 2 px	$>$ 3 px	$>$ 5 px
Naive approach	0.0555	0.0733	0.0471	0.0296
Early fusion	0.0644	0.0667	0.0526	0.0398
Our method	**0.0468**	**0.0393**	**0.0276**	**0.0201**

convolution layer.

7.5.3 Generalizing to Other Datasets

To illustrate that our method can generalize to other datasets, we compare our method to several methods on the Synthia dataset [146]. Synthia contains 5 sequences under different scenarios. And for each scenario, they capture images under different lighting and weather conditions such as Spring, Winter, Soft-rain, Fog and Night. We show quantitative results of experiments in Table. 7.5 and qualitative results are provided in the supplementary materials.

For sparse disparity inputs, we randomly selected 10% of full image resolution. As discussed before, projected Lidar points have misalignment with stereo images in KITTI dataset. To simulate the similar interference, we add various density levels of Gaussian noise to sparse disparity maps. As shown in 7.7, our proposed LidarStereoNet adapts well to the noisy input disparity maps, while S2D [124] fail completely to recover disparity information.

Table 7.5: **Quantitative results on the Synthia dataset.**

Methods	Abs Rel	$>$ 2 px	$>$ 3 px	$>$ 5 px
SPS-ST [131]	0.0475	0.0980	0.0879	0.0713
S2D [124]	0.0864	0.5287	0.4414	0.270
SINet [35]	**0.0290**	0.0642	0.0472	**0.0283**
Our method	0.0334	**0.0446**	**0.0373**	0.0299

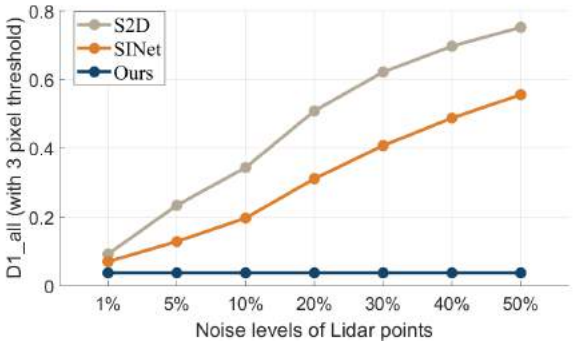


Figure 7.7: **Ablation study on noise resistance on Synthia dataset.** Our method has a consistent performance while the others have a notable performance drop.

7.6 Conclusion

In this chapter, we have proposed an unsupervised end-to-end learning based Lidar-Stereo fusion network "LidarStereoNet" for accurate 3D perception in real world scenarios. To effectively handle the noise in Lidar points and the potential misalignment between sensors, we present a novel "Feedback Loop" to select clean measurements by comparing output stereo disparities and input Lidar points. We have also introduced a piecewise slanted plane fitting loss into our network, thus enforcing strong 3D structure regularization on generated disparity maps. Our LidarStereoNet does not need ground-truth disparity maps for training and has good generalization capabilities. Extensive experiments on both real and synthetic datasets prove the superiority of our approach, which clearly outperforms state-of-the-art stereo matching and depth completion works with a large margin. Our approach could reliably work even when Lidar points are completely missing. In the future, we plan to extend our method to other depth perception and sensor fusion scenarios.

Part IV

Learning depth from a single image

Chapter 8

Stereo computation from a mixture image

This chapter proposes an original problem of *stereo computation from a single mixture image*– a challenging problem that had not been researched before. The goal is to separate (*i.e.*, unmix) a single mixture image into two constituted image layers, such that the two layers form a left-right stereo image pair, from which a valid disparity map can be recovered. This is a severely illposed problem, from one input image one effectively aims to recover three (*i.e.*, left image, right image and a disparity map). In this work we give a novel deep-learning based solution, by jointly solving the two subtasks of image layer separation as well as stereo matching. Training our deep net is a simple task, as it does not need to have disparity maps. Extensive experiments demonstrate the efficacy of our method [1].

8.1 Introduction

Stereo computation (stereo matching) is a well-known and fundamental vision problem, in which a dense depth map D is estimated from two images of the scene from slightly different viewpoints. Typically, one of the cameras is in the left (denoted by I_L) and the other in the right (denoted by I_R), just like we have left and right two eyes. Given a single image, it is generally impossible to infer a disparity map, unless using strong semantic-dependent image priors such as those single-image depth-map regression works powered by deep-learning [112, 109, 207]. Even though these learning based monocular depth estimation methods could predict a reasonable disparity map from a single image, they all assume the input image to be *an original colour image*.

In this chapter, we propose a novel and original problem, assuming instead one is provided with one *single mixture image* (denoted by I) which is a composition of an original stereo image pair I_L and I_R, *i.e.*, $I = f(I_L, I_R)$, and the task is to simultaneously recover both the stereo image pair I_L and I_R, and an accurate dense depth-map D. Under our problem definition, f denotes different image composition

[1] This work was originally published in [206]

operators that generate the mixture image, which is to be defined in details later. This is a very challenging problem, due to the obvious ill-posed (under-constrained) nature of the task, namely, from one input mixture image I one effectively wants to recover three images (I_L, I_R, and D).

In theory it appears to be a blind signal separation (BSS) task, *i.e.*, separating an image into two different component images. However, conventional methods such as BSS using independent component analysis (ICA) [59] are unsuitable for this problem as they make strong assumptions on the statistical independence between the two components. Under our problem definition, I_L, I_R are highly correlated. In computer vision, image layer separation such as reflection and highlight removal [60, 61] are also based on the difference in image statistics, again, unsuitable. Another related topic is image matting [208], which refers to the process of accurate foreground estimation from an image. However it either needs human interaction or depends on the difference between foreground object and background, which cannot be applied to our task.

In this chapter, we advocate a novel deep-learning based solution to the above task, by using a simple network architecture. We could successfully solve for a stereo pair L, R and a dense depth map D from a single mixture image I. Our network consists of an image separation module and a stereo matching module, where the two modules are optimized jointly. Under our framework, the solution of one module benefits the solution of the other module. It is worth-noting that the training of our network does not require ground truth depth maps.

At a first glance, this problem while intrigue, has pure intellectual interest only, not perhaps no practical use. In contrast, we show this is not the case: in this chapter, we show how to use it to solve for three very different vision problems: double vision, de-anaglyphy and even monocular depth estimation.

The requirement for de-anaglyph is still significant. If search on Youtube, there are hundreds if not thousands of thousands of anaglyph videos, where the original stereo images are not necessarily available. Our methods and the previous work [63][62] enable the recovery of the stereo images and the corresponding disparity map, which will significantly improve the users' *real 3D experience*. As evidenced in the experiments, our proposed method clearly outperforms the existing work with a wide gap. Last but not least, our model could also handle the task of monocular depth estimation and it comes as a surprise to us: Even with one single mixture image, trained on the KITTI benchmark, our method produces the state of the art depth estimation, with results much better than those traditional two images based methods.

8.2 Setup the stage

In this chapter we study two special cases of our novel problem of *joint image separation and stereo computation*, namely *anaglyph* (red-cyan stereo) and *diplopia* (double vision) (see Figure. 8.1), which have not been well-studied in the past.

Figure 8.1: Examples of image separation for single image based stereo computation. **Left column:** A double-vision image is displayed here. **Right column:** A red-cyan stereo image contains channels from the left and right images.

1) **Double vision (aka. diplopia)**: *Double vision* is the simultaneous perception of two images (a stereo pair) of a single object in the form of a single mixture image. Specifically, under the double vision (diplopia) model (*cf.* Fig. 8.1 (left column)), the perceived image $I = f(I_L, I_R) = (I_L + I_R)/2$, *i.e.*, the image composition is f a direct average of the left and the right images. Note that the above equation shares similarity with the linear additive model in layer separation [60, 209, 210] for reflection removal and raindrop removal, we will discuss the differences in details later.
2) **Red-Cyan stereo (aka. anaglyph)**: An anaglyph (*cf.* Fig. 8.1 (right column)) is a single image created by selecting chromatically opposite colours (typically red and cyan) from a stereo pair. Thus given a stereo pair I_L, I_R, the image composition operator f is defined as $I = f(I_L, I_R)$, where the red channel of I is extracted from the red channel of I_L while its green and blue channels are extracted from I_R. De-anaglyph [63, 62] aims at estimating both the stereo pair I_L, I_R (colour restoration) and computing its disparity maps.

At a first glance, the problem seems impossible as one has to generate two images plus a dense disparity map from *one single input*. However, since the two constitute images are not arbitrary but related by a valid disparity map. Therefore, they must be able to aligned well along the scanlines horizontally. For anaglyph stereo, existing methods [63, 62] exploit both image separation constraint and disparity map computation to achieve colour restoration and stereo computation. Joulin and Kang [62] reconstructed the original stereo pairs given the input anaglyph by using a modified SIFT-flow method [211]. Williem *et al.* [63] presented a method to solve the problem within iterations of colour restoration and stereo computation. These works suggest that by properly exploiting the image separation and stereo constraints, it is possible to restore the stereo pair images and compute the disparity map from a single mixture image.

There is little work in computer vision dealing with double vision (diplopia), which is nonetheless an important topic in ophthalmology and visual cognition. The most related works seem to be layer separation [60, 209], where the task is to decompose an input image into two layers corresponding to the background image and the foreground image. However, there are significant differences between our problem and general layer separation. For layer separation, the two layers of the compos-

ited image are generally independent and statistically different. In contrast, the two component images are highly correlated for double vision.

Even though there have been remarkable progresses in monocular depth estimation, current state-of-the-art network architectures [112, 109] and [113] cannot be directly applied to our problem. This is because that they depend on a single left-/right image input, which is unable to handle image mixture case investigated in this work. Under our problem definition, the two tasks of image separation and stereo computation are tightly coupled: stereo computation is not possible without correct image separation; on the other hand, image separation will benefit from disparity computation.

In this chapter, we present a unified framework to handle the problem of stereo computation for a single mixture image, which naturally unifies various geometric vision problems such as anaglyph, diplopia and even monocular depth estimation. Our network can be trained with the supervision of stereo pair images only without the need for ground truth disparity maps, which significantly reduces the requirements for training data. Extensive experiments demonstrate that our method achieves superior performances.

8.3 Our Method

In this chapter, we propose an end-to-end deep neural network to simultaneously learn image separation and stereo computation from a single mixture image. It can handle a variety forms of problems such as anaglyph, de-diplopia and even monocular depth estimation. Note that existing work designed for either layer-separation or stereo-computation cannot be applied to our problem directly. This is because these two problems are deeply coupled, *i.e.,* the solution of one problem affects the solution of the other problem. By contrast, our formulation to be presented as below, jointly solves both problems.

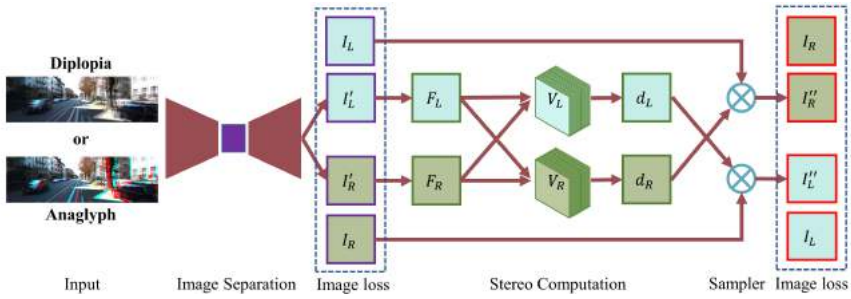


Figure 8.2: **Overview of our proposed stereo computation for a single mixture image framework.** Our network consists of an image separation module and a stereo computation module. Take a single mixture image as input, our network simultaneously separates the image into a stereo image pair and computes a dense disparity map.

8.3.1 Mathematical Formulation

Under our mixture model, quality of depth map estimation and image separation are evaluated jointly and therefore, the solution of each task can benefit from each other. Our network model (*cf.* , Fig. 8.2) consists of two modules, *i.e.,* an image separation module and a stereo computation module. During network training, only the ground-truth stereo pairs are needed to provide supervisions for both image separation and stereo computation.

By considering both the image separation constraint and the stereo computation constraint in network learning, we define the overall loss function as:

$$\mathcal{L}(\theta_L, \theta_R, \theta_D) = \mathcal{L}_C(\theta_L, \theta_R) + \mathcal{L}_D(\theta_D), \tag{8.1}$$

where $\theta_L, \theta_R, \theta_D$ denote the network parameters corresponding to the image separation module (left image prediction and right image prediction) and the stereo computation module. A joint optimization of $(\theta_L, \theta_R, \theta_D) = \arg\min \mathcal{L}(\theta_L, \theta_R, \theta_D)$ gives both the desired stereo image pair and the disparity map.

8.3.2 Image Separation

The input single mixture image $I \in \mathbb{R}^{H \times W \times 3}$ encodes the stereo pair image as $I = f(I_L, I_R)$, where f is the image composition operator known a priori. To learn the stereo image pair from the input single mixture image, we present a unified end-to-end network pipeline. Specifically, denote $\mathcal{F}$ as the learned mapping from the mixture image to the predicted left or right image parameterized by θ_L or θ_R. The objective function of our image separation module is defined as,

$$\alpha_c \mathcal{L}_c(\mathcal{F}(I; \theta_L), I_L) + \alpha_p \mathcal{L}_p(\mathcal{F}(I; \theta_L)), \tag{8.2}$$

where I is the input single mixture image, I_L, I_R are the ground truth stereo image pair. The loss function $\mathcal{L}$ measures the discrepancy between the predicted stereo images and the ground truth stereo images. The object function for the right image is defined similarly.

In evaluating the discrepancy between images, various loss functions such as ℓ_2 loss [212], classification loss [10] and adversarial loss [213] can be applied. Here, we leverage the pixel-wise ℓ_1 regression loss as the content loss of our image separation network,

$$\mathcal{L}_c(\mathcal{F}(I; \theta_L), I_L) = \left| \mathcal{F}(I; \theta_L) - I_L \right|. \tag{8.3}$$

This loss allows us to perform end-to-end learning as compatible with the stereo matching loss and do not need to consider class imbalance problem or add an extra network structure as a discriminator.

Researches on natural image statistics show that a typical real image obeys sparse spatial gradient distributions [214]. According to Yang *et al.* [60], such a prior can be represented as the Total Variation (TV) term in energy minimization. Therefore, we

have our image prior loss:

$$\mathcal{L}_p(\mathcal{F}(I;\theta_L)) = |\mathcal{F}(I;\theta_L)|_{\mathrm{TV}} = |\nabla\mathcal{F}(I;\theta_L)|\,, \tag{8.4}$$

where ∇ is the gradient operator.

We design a U-Net architecture [154] for image separation, which has been used in various conditional generation tasks. Our image separation module consists of 22 convolutional layers. Each convolutional layer contains one convolution-relu pair except for the last layer and we use element-wise add for each skip connection to accelerate the convergence. For the output layer, we utilize a "tanh" activation function to map the intensity value between -1 and 1. A detailed description of our network structure is provided in the supplemental material.

The output of our image separation module is a 6 channels image, where the first 3 channels represent the estimated left image $\mathcal{F}(I;\theta_L)$ and the rest 3 channels for the estimated right image $\mathcal{F}(I;\theta_R)$. When the network converges, we could directly use these images as the image separation results. However, for the de-anaglyph task, as there is extra constraint (the mixture happens at channel level), we could leverage the colour prior of an anaglyph that the desired image separation (colourization) can be further improved by warping corresponding channels based on the estimated disparity maps.

For the monocular depth estimation task, only the right image will be needed as the left image has been provided as input.

8.3.3 Stereo Computation

The input to the stereo computation module is the separated stereo image pair from the image separation module. The supervision of this module is the ground truth stereo pairs rather than the inputs. The benefit of using ground truth stereo pairs for supervision is that it makes the network not only learn how to find the matching points, but also makes the network to extract features that are robust to the noise from the generated stereo images.

Fig. 8.2 shows an overview of our stereo computation architecture, we adopt a similar stereo matching architecture from Zhong *et al.* [14] without its consistency check module. The benefit for choosing such a structure is that their model can converge within 2000 iterations which makes it possible to train the entire network in an end-to-end fashion. Additionally, removing the need of ground truth disparity maps enables us to access much more accessible stereo images.

Our loss function for stereo computation is defined as:

$$\mathcal{L}_D = \omega_w(\mathcal{L}_w^l + \mathcal{L}_w^r) + \omega_s(\mathcal{L}_s^l + \mathcal{L}_s^r), \tag{8.5}$$

where $\mathcal{L}_w^l, \mathcal{L}_w^r$ denote the image warping appearance loss, $\mathcal{L}_s^l, \mathcal{L}_s^r$ express the smoothness constraint on the disparity map.

Similar to $\mathcal{L}_c$, we form a loss in evaluating the image similarity by computing the pixel-wise ℓ_1 distance between images. We also add a structural similarity term

SSIM [143] to improve the robustness against illumination changes across images. The appearance loss $\mathcal{L}_w^l$ is derived as:

$$\mathcal{L}_w^l(I_L, I_L^{''}) = \frac{1}{N}\sum \lambda_1 \frac{1 - \mathcal{S}(I_L, I_L^{''})}{2} + \lambda_2 \left| I_L - I_L^{''} \right|, \tag{8.6}$$

where N is the total number of pixels and $I_L^{''}$ is the reconstructed left image. λ_1, λ_2 balance between structural similarity and image appearance difference. According to [109], $I_L^{''}$ can be fully differentially reconstructed from the right image I_R and the right disparity map d_R through bilinear sampling [215].

For the smoothness term, similar to [109], we leverage the Total Variation (TV) and weigh it with image's gradients. Our smoothness loss for disparity field is:

$$\mathcal{L}_s^l = \frac{1}{N}\sum |\nabla_u d_L|\, e^{-|\nabla_u I_L|} + |\nabla_v d_L|\, e^{-|\nabla_v I_L|}. \tag{8.7}$$

8.3.4 Implementation Details

We implement our network in TensorFlow [216] with 17.1M trainable parameters. Our network can be trained from scratch in an end-to-end fashion with a supervision of stereo pairs and optimized using RMSProp [217] with an initial learning rate of 1×10^{-4}. Input images are normalized with pixel intensities level ranging from -1 to 1. For the KITTI dataset, the input images are randomly cropped to 256×512, while for the Middlebury dataset, we use 384×384. We set disparity level to 96 for the stereo computation module. For weighting loss components, we use $\alpha_c = 1, \alpha_p = 0.2, \omega_w = 1, \omega_s = 0.05$. We set $\lambda_1 = 0.85, \lambda_2 = 0.15$ throughout our experiments. Due to the hardware limitation (Nvidia Titan Xp), we only use batch size 1 during network training.

8.4 Experiments and Results

In this section, we validate our proposed method and present experimental evaluation for both de-anaglyph and de-diplopia (double vision). For experiments on anaglyph images, given a pair of stereo images, the corresponding anaglyph image can be generated by combining the red channel of the left image and the green/blue channels of the right image. Any stereo pairs can be used to quantitatively evaluate the performance of de-anaglyph. However, since we also need to quantitatively evaluate the performance of anaglyph stereo matching, we use two stereo matching benchmarking datasets for evaluation: Middlebury dataset [43] and KITTI stereo 2015 [90]. Our network is initially trained on the KITTI Raw dataset with 29000 stereo pairs that listed by [109] and further fine-tuned on Middlebury dataset. To highlight the generalization ability of our network, we also perform qualitative experiments on random images from Internet. For de-diplopia (double vision), we synthesize our inputs by averaging stereo pairs. Qualitative and quantitative results are reported on

KITTI stereo 2015 benchmark [90] as well. Similar to the de-anaglyph experiment, we train our initial model on the KITTI raw dataset.

8.4.1 Advantages of joint optimization

Our framework consists of image separation and stereo computation, where the solution of one subtask benefits the solution of the other subtask. Direct stereo computation is impossible for a single mixture image. To analyze the advantage of joint optimization, we perform ablation study in image separation without stereo computation and the results are reported in Table 8.1. Through joint optimization, the average PSNR increases from 19.5009 to 20.0914, which demonstrates the benefit of introducing the stereo matching loss in image separation.

Table 8.1: Ablation study of image separation on KITTI.

Metric	Image separation only	Joint optimization
PSNR	19.5009	20.0914

8.4.2 Evaluation of Anaglyph Stereo

We compare the performance of our method with two state-of-the-art de-anaglyph methods: Joulin *et al.* [62] and Williem *et al.* [63]. Evaluations are performed on two subtasks: stereo computation and image separation (colour restoration).
Stereo Computation. We present qualitative comparison of estimated disparity maps in Fig. 8.3 for Middlebury [43] and in Fig. 8.4 for KITTI 2015 [90]. Stereo pairs in Middlebury are indoor scenes with multiple handcrafted layouts and the ground truth disparities are captured by highly accurate structural light sensors. On the other hand, the KITTI stereo 2015 consists of 200 outdoor frames in their training set, which is more challenging than the Middlebury dataset. The ground truth disparity maps are generated by sparse LIDAR points and CAD models.

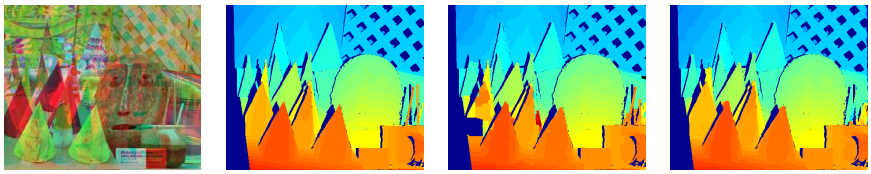


Figure 8.3: Qualitative stereo computation results on the Middlebury dataset by our method. From left to right: input anaglyph image, ground truth disparity map, disparity map generated by Williem *et al.* [63] and our method.

On both datasets, our method can generate more accurate disparity maps than previous ones from visual inspection. It can be further evidenced by the quantitative results of bad pixel percentage that shown in Table. 8.2 and Fig. 8.5. For the Middlebury dataset, our method achieves 32.55% performance leap than Williem *et al.* [63]

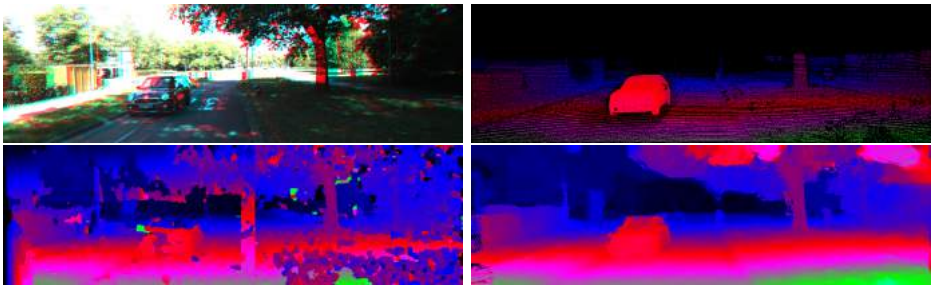


Figure 8.4: Qualitative disparity map recovery results on KITTI-2015 of our method. Top row: Input anaglyph image and ground truth disparity map. Bottom row: Result of Williem *et al.* [63] and our result.

and 352.28% performance leap than Joulin *et al.* [62]. This is reasonable as Joulin *et al.* [62] did not add disparity into its optimization. For the KITTI dataset, we achieve an average bad pixel ratio (denoted as D1_all) of 5.96% with 3 pixel thresholding across 200 images in the training set as opposed to 13.66% by Joulin *et al.* [62] and 14.40% by Williem *et al.* [63].

Table 8.2: Performance comparison in disparity map estimation for de-anaglyph on the Middlebury dataset. We report the bad pixel ratio with a threshold of 1 pixel. Disparities are scaled according to the provided scaling factor on the Middlebury dataset.

Method	Tsukuba	Venus	Cones	Teddy	Mean
Joulin [62]	14.02	25.90	23.49	43.85	26.82
Williem [63]	12.53	2.24	8.00	8.68	7.86
Ours	**10.44**	**1.25**	**4.51**	**7.51**	**5.93**

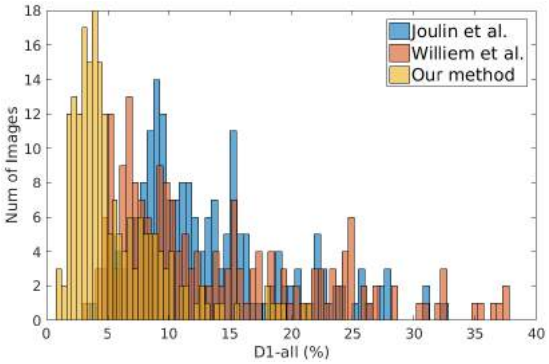


Figure 8.5: Disparity map estimation results comparison on the KITTI stereo 2015 dataset.

Image Separation. As an anaglyph image is generated by concatenating the red channel from the left image and the green and blue channels from the right image, the original colour can be found by warping the corresponding channels based on the estimated disparity maps. We leverage such a prior for de-anaglyph and adopt the

Table 8.3: Performance comparisons (PSNR) in image separation (restoration) for the task of de-anaglyph on the Middlebury dataset.

Method	Tsukuba		Venus		Cones		Teddy	
	Left	Right	Left	Right	Left	Right	Left	Right
Joulin [62]	30.83	32.88	29.66	31.97	21.52	24.54	21.16	24.59
Williem [63]	30.32	31.12	31.41	34.93	23.68	26.17	23.14	31.05
Ours	**32.99**	**35.31**	**35.05**	**37.74**	**26.31**	**30.17**	**28.44**	**35.53**

post-processing step from Joulin *et al.* [62] to handle occluded regions. Qualitative and quantitative comparison of image separation performance are conducted on the Middlebury and KITTI datasets. We employ the Peak Signal-to-Noise Ratio (PSNR) to measure the image restoration quality.

Qualitative results for both datasets are provided in Fig. 8.6 and Fig. 8.7. Our method is able to recover colours in the regions where ambiguous colourization options exist as those areas rely more on the correspondence estimation, while other methods tend to fail in this case.

Figure 8.6: Qualitative image separation results on the KITTI-2015 dataset. Top to bottom: Input, ground truth, result from Williem *et al.* [63], our result. Our method successfully recovers the correct colour of the large textureless region on the right of the image while the other method fails.

Table 8.3 and Table 8.4 report the performance comparison between our method and state-of-the-art de-anaglyph colourization methods: Joulin *et al.* [62] and Williem *et al.* [63] on the Middlebury dataset and on the KITTI dataset correspondingly. For the KITTI dataset, we calculated the mean PSNR throughout the total 200 images of the training set. Our method outperforms others with a notable margin. Joulin *et al.* [62] is able to recover relatively good restoration results when the disparity level is small, such as Tsukuba, Venus, and KITTI. When the disparity level doubled, its performance drops quickly as for Cone and Teddy images. Different with Williem *et al.* [63], which can only generate disparity maps at pixel level, our method is able to further optimize the disparity map to sub-pixel level, therefore achieves superior performance in both stereo computation and image restoration (separation).

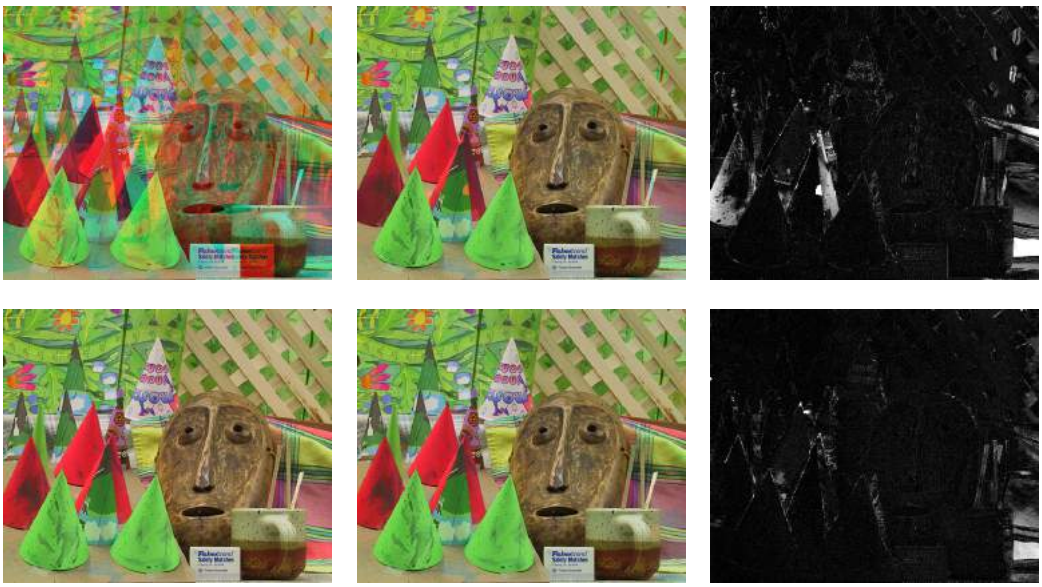


Figure 8.7: Qualitative comparison in image separation (restoration) on the Middlebury dataset. The first column shows the input anaglyph image and the ground truth image. The results of Williem *et al.* [63] (top) and our method (bottom) with their corresponding error maps are shown in the second and the third column.

Table 8.4: Performance comparisons (PSNR) in image separation (restoration) for the task of de-anaglyph on the KITTI dataset.

Dataset	View	Joulin [62]	Williem [63]	Ours
KITTI	Left	25.13	24.57	**26.30**
	Right	27.19	26.94	**28.76**

Anaglyph in the wild. One of the advantages of conventional methods is their generalization capability. They can be easily adapt to different scenarios with or without parameter changes. Deep learning based methods, on the other hand, are more likely to have a bias on specific dataset. In this section, we provide qualitative evaluation of our method on anaglyph images downloaded from the Internet to illustrate the generalization capability of our method. Our method, even though trained on the KITTI dataset which is quite different from all these images, achieves reliable image separation results as demonstrated in Fig. 8.8. This further confirms the generalization ability of our network model.

8.4.3 Evaluation for double-vision unmixing

Here, we evaluate our proposed method for unmixing of double-vision image, where the input image is the average of a stereo pair. Similar to anaglyph, we evaluate our performance based on the estimated disparities and reconstructed stereo pair on

Figure 8.8: Qualitative stereo computation and image separation results for real world anaglyph images downloaded from the Internet. Left to right: input anaglyph images, disparity maps of our method, image separation results of our method.

the KITTI stereo 2015 dataset. For disparity evaluation, we use the oracle disparity maps (that are computed with clean stereo pairs) as a reference in Fig. 8.9. The mean bad pixel ratio of our method is 6.67%, which is comparable with the oracle's performance as 5.28%. For image separation, we take a layer separation method [209] as a reference. A quantitative comparison is shown in Table 8.5. Conventional layer separation methods tend to fail in this scenario as the statistic difference between the two mixed images is minor which violates the assumption of these methods. Qualitative results of our method are shown in Fig. 8.10.

Table 8.5: Performance comparison in term of PSNR for the task of image restoration from double-vision images on the KITTI dataset.

Dataset	View	Li *et al.* [209]	Ours
KITTI	Left	17.03	**24.38**
	Right	7.67	**24.55**

8.5 Beyond Anaglyph and Double-Vision

Our problem definition also covers the problem of monocular depth estimation, which aims at estimating a depth map from a single image [65, 109, 218, 207]. Under this setup, the image composition operator f is defined as $I = f(I_L, I_R) = I_L$ or $I = f(I_L, I_R) = I_R$, *i.e.*, the mixture image is the left image or the right image. Thus, monocular depth estimation is a special case of our problem definition.

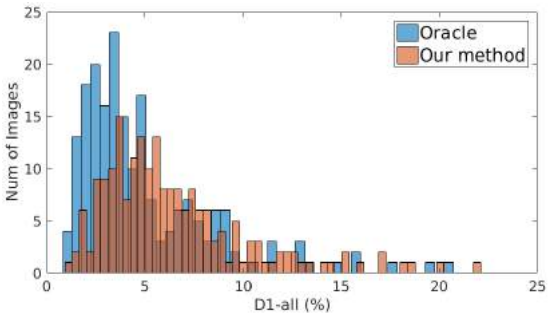


Figure 8.9: Stereo computation results on the KITTI 2015 dataset. The oracle disparity map is computed with clean stereo images.

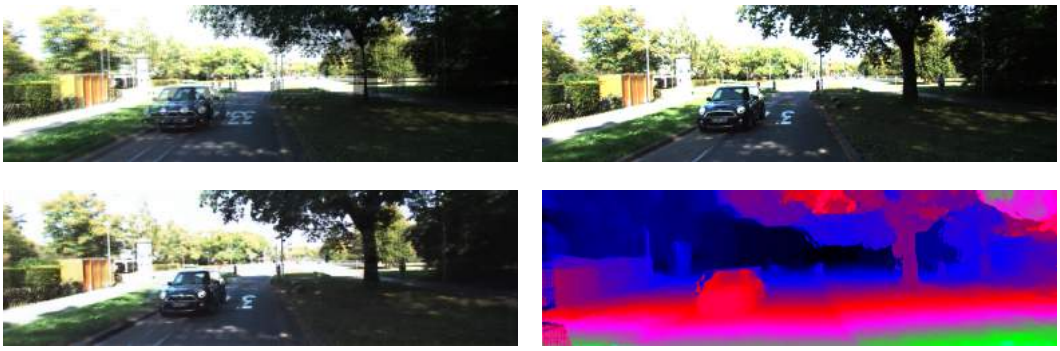


Figure 8.10: Qualitative diplopia unmixing results by our proposed method. Top to bottom,left to right: input diplopia image, ground truth left image, restored left image by our method, and our estimated disparity map.

We evaluated our framework for monocular depth estimation on the KITTI 2015 dataset. Quantitative results and qualitative results are provided in Table 8.6 and Fig. 8.11, where we compare our method with state-of-the-art methods [112], [97] and [109]. Our method, even designed for a much more general problem, outperforms both [112] and [97] and achieves quite comparable results with [109].

8.6 Conclusion

This chapter has defined a novel problem of stereo computation from a single mixture image, where the goal is to separate a single mixture image into a pair of stereo images–from which a legitimate disparity map can be estimated. This problem definition naturally unifies a family of challenging and practical problems such as anaglyph, diplopia and monocular depth estimation. The problem goes beyond the scope of conventional image separation and stereo computation. We have presented a deep convolutional neural network based framework that jointly optimizes the image separation module and the stereo computation module. It is worth noting that

Table 8.6: Monocular depth estimation results on the KITTI 2015 dataset using the split of Eigen *et al.* [65]. Our model is trained on 22,600 stereo pairs from the KITTI raw dataset listed by [109] by 10 epochs. Depth metrics are from Eigen *et al.* [65]. Our performance is better than the state-of-the-art method [109].

Methods	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou *et al.* [97]	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Garg *et al.* [112]	0.169	1.080	5.104	0.273	0.740	0.904	0.962
Godard *et al.* [109]	0.140	0.976	4.471	0.232	0.818	0.931	0.969
Ours	0.126	0.835	3.971	0.207	0.845	0.942	0.975

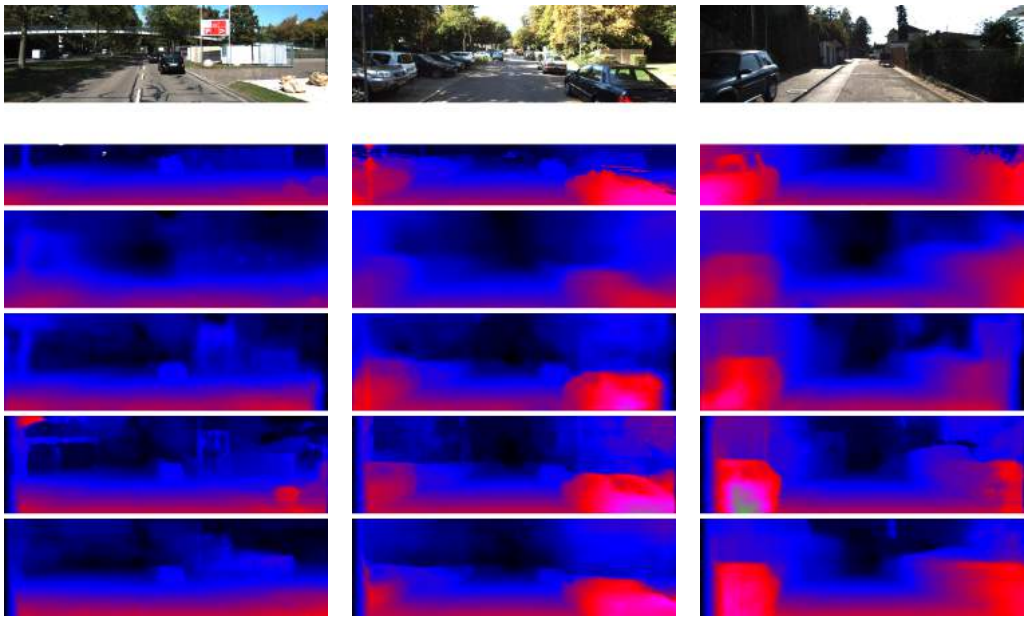


Figure 8.11: Qualitative monocular estimation evaluations on the KITTI-2015 dataset: Top to bottom: left image, ground truth, results from Zhou *et al.* [97], results from Garg *et al.* [112], results from Godard *et al.* [109] and our results. Since the ground truth depth points are very sparse, we interpolated it with a colour guided depth painting method [28] for better visualization.

we do not need ground truth disparity maps in network learning. In the future, we will explore additional problem setups such as "alpha-matting". Other issues such as occlusion handling and extension to handle video should also be considered.

Part V

Learning dense correspondence from a monocular camera

Chapter 9

Unsupervised Deep Epipolar Flow for Stationary or Dynamic Scenes

Unsupervised deep learning for optical flow computation has achieved promising results. Most existing deep-net based methods rely on image brightness consistency and local smoothness constraint to train the networks. Their performance degrades at regions where repetitive textures or occlusions occur. In this chapter, we propose Deep Epipolar Flow, an unsupervised optical flow method which incorporates global geometric constraints into network learning. In particular, we investigate multiple ways of enforcing the epipolar constraint in flow estimation. To alleviate a "chicken-and-egg" type of problem encountered in dynamic scenes where multiple motions may be present, we propose a low-rank constraint as well as a union-of-subspaces constraint for training. Experimental results on various benchmarking datasets show that our method achieves competitive performance compared with supervised methods and outperforms state-of-the-art unsupervised deep-learning methods [1].

9.1 Introduction

Optical flow estimation is a fundamental problem in computer vision with many applications. Since Horn and Schunck's seminal work [66], various methods have been developed using variational optimization [70, 71, 72], energy minimization [73, 74, 75, 76], or deep learning [77, 78, 79, 80]. In this paper, we particularly tackle the problem of unsupervised optical flow learning using deep convolutional neural networks (CNNs). Compared to its supervised counterpart, unsupervised flow learning does not require ground-truth flow, which is often hard to obtain, as supervision and can thus be applied in broader domains.

Recent research has been focused on transforming traditional domain knowledge of optical flow into deep learning, in terms of either training loss formulation or network architecture design. For example, in view of brightness consistency between two consecutive images, a constraint that has been commonly used in conventional optical flow methods, researchers have formulated photometric loss [110, 116], with

[1] This work was originally published in [16]

the help of fully differentiable image warping [219], to train deep neural networks. Other common techniques including image pyramid [220] (to handle large flow displacements), total variation regularization [221, 222] and occlusion handling [223] have also led to either new network structures (*e.g.,* pyramid networks [79, 80]) or losses (*e.g.,* smoothness loss and occlusion mask [117, 118]). In the unsupervised setting, existing methods mainly rely on the photometric loss and flow smoothness loss to train deep CNNs. This, however, poses challenges for the neural networks to learn optical flow accurately in regions with repetitive textures and occlusions. Although some methods [117, 118] jointly learn occlusion masks, these masks do not mean to provide more constraints but only to remove the outliers in the losses. In light of the difficulties of learning accurate flow in these regions, we propose to incorporate *global* epipolar constraints into flow network training in this paper.

Leveraging epipolar geometry in flow learning, however, is not a trivial task. An inaccurate or wrong estimate of fundamental matrices [92] would mislead the flow network training in a holistic way, and thus significantly degrade the model prediction accuracy. This is especially true when a scene contains multiple independent moving objects as one fundamental matrix can only describe the epipolar geometry of one rigid motion. Instead of posing a hard epipolar constraint, in this paper, we propose to use soft epipolar constraints that are derived using *low-rankness* when the scene is stationary, and *union of subspaces structure* when the scene is motion agnostic. We thus formulate corresponding losses to train our flow networks in an unsupervised manner.

Our work makes an attempt towards incorporating epipolar geometry into deep unsupervised optical flow computation. Through extensive evaluations on standard datasets, we show that our method achieves competitive performance compared with supervised methods, and outperforms existing unsupervised methods by a clear margin. Specifically, as of the date of paper submission, on KITTI and MPI Sintel benchmarks, our method achieves the best performance among published deep unsupervised optical flow methods.

9.2 Related work

Optical flow estimation has been extensively studied for decades. A significant number of papers have been published in this area. Below we only discuss a few geometry-aware methods and recent deep-learning based methods that we consider closely related to our method.

Supervised deep optical flow. Recently, end-to-end learning based deep optical flow approaches have shown their superiority in learning optical flow. Given a large amount of training samples, optical flow estimation is formulated to learn the regression between image pair and corresponding optical flow. These approaches achieve comparable performance to state-of-the-art conventional methods on several benchmarks while being significantly faster. FlowNet [77] is a pioneer in this direction, which needs a large-size synthetic dataset to supervise network learning. FlowNet2

[78] greatly extends FlowNet by stacking multiple encoder-decoder networks one after the other, which could achieve a comparable result to conventional methods on various benchmarks. Recently, PWC-Net [80] combines sophisticated conventional strategies such as pyramid, warping and cost volume into network design and set the state-of-the-art performance on KITTI [89, 90] and MPI Sintel [68]. These supervised deep optical flow methods are hampered by the need for large-scale training data with ground truth optical flow, which also limits their generalization ability.

Unsupervised deep optical flow. Instead of using ground truth flow as supervision, Yu *et al.* [110] and Ren *et al.* [111] suggested that, similar to conventional methods, the image warping loss can be used as supervision signals in learning optical flow. However, there is a significant performance gap between their work and the conventional ones. Then, Simon *et al.* [116] analyzed the problem and introduced bidirectional Census loss to handle illumination variation between frames robustly. Concurrently, Yang *et al.* [117] proposed an occlusion-aware warping loss to exclude occluded points in error computation. Very recently, Janai *et al.* [118] extended two-view optical flow to multi-view cases with improved occlusion handling performance. Introducing sophisticated occlusion estimation and warping loss reduces the performance gap between conventional methods and current unsupervised ones, nevertheless, the gap is still huge. To address this issue, we propose a global epipolar constraint in flow estimation that largely narrows the gap.

Geometry-aware optical flow. In the field of cooperating with geometry constrains, Valgaerts *et al.* [85] introduced a variational model to simultaneously estimate the fundamental matrix and the optical flow. Wedel *et al.* [86] utilized fundamental matrix prior as a weak constraint in a variational framework. Yamaguchi *et al.* [87] converted optical flow estimation task into a 1D search problem by using precomputed fundamental matrices and the small motion assumptions. These methods, however, assume that the scene is mostly rigid (and thus a single fundamental matrix is sufficient to constrain two-view geometry), and treat the dynamic parts as outliers [86]. Garg *et al.* [224] used the subspace constraint on multi-frame optical flow estimation as a regularization term. However, this approach, assumes an affine camera model and works over entire sequences. Wulff *et al.* [88] used semantic information to split the scene into dynamic objects and static background and only applied strong geometric constraints on the static background. Recently, inspired by multi-task learning, people started to jointly estimate depth, camera poses and optical flow in an unified framework [102, 99, 225]. These work mainly leverage a consistency between flows that estimated from a flow network and computed from poses and depth. This constraint only works for stationary scenes and their performance is only comparable with unsupervised deep optical flow methods.

By contrast, our proposed method is able to handle both stationary and dynamic scenarios without explicitly computing fundamental matrices. This is achieved by introducing soft epipolar constraints derived from epipolar geometry, using *low-rankness* and *union-of-subspaces* properties. Converting these constraints to proper losses, we can exert global geometric constraints in optical flow learning and obtain much better performance.

9.3 Epipolar Constraints in Optical Flow

Optical flow aims at finding dense correspondences between two consecutive frames. Formally, let I^t denote the image at time t, and I^{t+1} the next image. For pixels $\mathbf{x}_i$ in I^t, we would like to find their correspondences $\mathbf{x}'_i$ in I^{t+1}. The displacement vectors $\mathbf{v} = [\mathbf{v}_1, ..., \mathbf{v}_N] \in \mathbb{R}^{2\times N}$ (with N the total number of pixels in I^t) are the optical flow we would like to estimate.

Recall that in two-view epipolar geometry [92], by using the homogeneous coordinates, a pair of point correspondences in two frames $\mathbf{x}'_i = (x'_i, y'_i, 1)^T$ and $\mathbf{x}_i = (x_i, y_i, 1)^T$ is related by a fundamental matrix $\mathbf{F}$,

$$\mathbf{x}_i'^T \mathbf{F} \mathbf{x}_i = 0 \,. \tag{9.1}$$

In the following sections, we show how to enforce the epipolar constraint as a global regularizer in flow learning.

9.3.1 Two-view Geometric Constraint

Given estimated optical flow $\mathbf{v}$, we can convert it to a series of correspondences $\mathbf{x}_i$ and $\mathbf{x}'_i$ in I^t and I^{t+1} respectively. Then these corresponding points can be used to compute a fundamental matrix $\mathbf{F}$ by normalized 8 points method [92]. Once the $\mathbf{F}$ is estimated, we can compute its fitting error. Directly optimizing Eq. (9.1) is not effective as it is only an algebraic error that does not reflect the real geometric distances. We can use the Gold Standard method [92] to compute the geometric distances but it requires reconstructing the 3D points $\widehat{\mathbf{X}}_i$ beforehand for every point. Otherwise, we can use its first-order approximation, the Sampson distance $\mathcal{L}_\mathbf{F}$ to represent the geometric error,

$$\mathcal{L}_\mathbf{F} = \sum_i^N \frac{(\mathbf{x}_i'^T \mathbf{F} \mathbf{x}_i)^2}{(\mathbf{F}\mathbf{x}_i)_1^2 + (\mathbf{F}\mathbf{x}_i)_2^2 + (\mathbf{F}^T\mathbf{x}'_i)_1^2 + (\mathbf{F}^T\mathbf{x}'_i)_2^2} \,. \tag{9.2}$$

The difficulty of optimizing this equation comes from its *chicken-and-egg* character: it consists of two mutually interlocked sub-problems, *i.e., estimating a fundamental matrix* $\mathbf{F}$ *from an estimated flow* and *updating the flow to comply with the* $\mathbf{F}$. This alternating method, therefore, heavily relies on proper initialization.

Up to now, we have only considered the static scene scenario, where only egomotion exists. In a multi-motion scene, this method requires estimating $\mathbf{F}$ for each motion, which again needs a motion segmentation step. It is still feasible to address this problem via iteratively solving three sub-tasks: (i) *update flow estimation*; (ii) *estimate* $\mathbf{F}^m$ *for each rigid motion given current motion segmentation*; (iii) *update motion segmentation based on the nearest* $\mathbf{F}^m$.

However, this method again has several inherent limitations. First, the number of motions need to be known as a priori which is almost impossible in general optical flow estimation. Second, this method is still sensitive to the quality of initial optical flow estimation and motion labels. Incorrectly estimated flow may generate wrong

$\mathbf{F}^m$, which will in turn lead flow estimation to the wrong solution, therefore making the estimation even worse. Third, the motion segmentation step is non-differentiable, so with it, an end-to-end learning becomes impossible.

To overcome these drawbacks, we formulate two soft epipolar constraints using *low-rankness* and *union-of-subspaces* properties. And we will show that these constraints can be easily included as extra losses to regularize the network learning.

9.3.2 Low-rank Constraint

In this section, we show that it is possible to enforce a soft epipolar constraint without explicitly computing the fundamental matrix in a static scene.

Note that we can rewrite the epipolar constraint in Eq. (9.1) as

$$\mathbf{f}^T \text{vec}(\mathbf{x}'_i \mathbf{x}_i^T) = 0 \,, \tag{9.3}$$

where $\mathbf{f} \in \mathbb{R}^9$ is the vectorized fundamental matrix of $\mathbf{F}$ and

$$\text{vec}(\mathbf{x}'_i \mathbf{x}_i^T) = (x_i x'_i, x_i y'_i, x_i, y_i x'_i, y_i y'_i, y_i, x'_i, y'_i, 1)^T \,. \tag{9.4}$$

Observe that, $\text{vec}(\mathbf{x}'_i \mathbf{x}_i^T)$ lies on a subspace (of dimension up to eight), called epipolar subspace [226]. Let us define $\mathbf{h}_i = \text{vec}(\mathbf{x}'_i \mathbf{x}_i^T)$. Then the data matrix $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_N]$ should be low-rank. This provides a possible way of regularizing optical flow estimation via rank minimization instead of explicitly computing $\mathbf{F}$. Specifically, we can formulate a loss as

$$\mathcal{L}_{\text{lowrank}} = \text{rank}(\mathbf{H}) \,, \tag{9.5}$$

which is unfortunately non-differentiable and is thus not feasible to serve as a loss for flow network training. Fortunately, we can still use its convex surrogate, the nuclear norm, to form a loss as

$$\mathcal{L}^*_{\text{lowrank}} = \|\mathbf{H}\|_* \,, \tag{9.6}$$

where the nuclear norm $||\cdot||_*$ can be computed by performing singular value decomposition (SVD) of $\mathbf{H}$. Note that the SVD operation is differentiable and has been implemented in modern deep learning toolboxes such as Tensorflow and Pytorch, so this nuclear norm loss can be easily incorporated into network training. We also note that though this low-rank constraint is derived from epipolar geometry described by a fundamental matrix, it still applies in degenerate cases where a fundamental matrix does not exist. For example, when the motion is all zero or pure rotational, or the scene is fully planar, $\mathbf{H}$ will have rank six; under certain special motions, *e.g.,* an object moving parallel to the image plane, its $\mathbf{H}$ will have rank seven.

Comparing to the original epipolar constraint, one may concern that this low-rank constraint is too loose to be effective, especially when the ambient space dimension is only nine. Although a thorough theoretical analysis is out of the scope of this paper (interested readers may refer to literature such as [227]), we will show in our experiments that this loss can improve the model performance by a significant margin when trained on data with mostly static scenes. However, this loss becomes

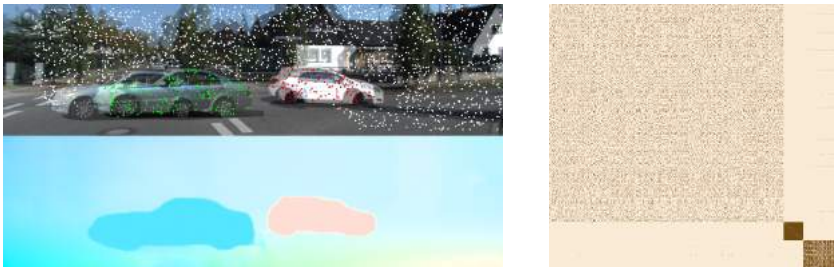


Figure 9.1: **Motion segmentation and affinity matrix (constructed from C) visualization.** The scene contains three motions annotated by three different colours: the ego-motion and the two cars' movements. On the right, we show a constructed affinity matrix from $\mathbf{C}$ which contains three diagonal blocks corresponding to these three motions. On the bottom left, we illustrate our estimated optical flow and the top left image shows that all these three motions are correctly segmented based on the $\mathbf{C}$. The sparse dots on the image are the sampled 2000 points that has been used to compute $\mathbf{C}$. It proves that our Union-of-Subspace constraint can work under multi-body scenarios.

ineffective when a scene has more than one motion, as the matrix $\mathbf{H}$ will then be full-rank.

9.3.3 Union-of-Subspaces Constraint

In this section, we introduce another soft epipolar constraint, namely *union-of-subspaces constraint*, which can be applied in broader cases.

From Eq. (9.4), it's not hard to observe that $\mathbf{h}_i$ from one rigid motion lies on a common epipolar subspace because they all share the same fundamental matrix. When there are multiple motions in a scene, $\mathbf{h}_i$ will lie in a union of subspaces. Note that this union-of-subspace structure has been shown to be useful in motion segmentation from two perspective images [228]. Here, we re-formulate it in optical flow learning and come up with an effective loss using closed-form solutions.

In particular, the union-of-subspaces structure can be characterized by the self-expressiveness property [229], *i.e.*, a data point in one subspace can be represented by a linear combination of other points from the same subspace. This has translated into a mathematical optimization problem [230, 231] as

$$\min_{\mathbf{C}} \frac{1}{2}\|\mathbf{C}\|_F^2 \quad \text{s.t.} \quad \mathbf{H} = \mathbf{H}\mathbf{C}\,. \tag{9.7}$$

where $\mathbf{C}$ is the subspace self-expression coefficient and $\mathbf{H}$ is a matrix function of estimated flows. Note that, in subspace clustering literature, other norms on $\mathbf{C}$ have also been used, *e.g.*, nuclear norm in [232] and ℓ_1 norm in [229]. We are particularly interested in the Frobenius norm regularization due to its simplicity and equivalence to nuclear norm optimization [231], which is crucial for formulating an effective loss for CNN training.

However, in real world scenarios, the flow estimation inevitably contains noises. Therefore, we relax the constraints in Eq.(9.7) by alternatively optimizing the function below

$$\mathcal{L}_{\text{subspace}} = \frac{1}{2}\|\mathbf{C}\|_F^2 + \frac{\lambda}{2}\|\mathbf{H}\mathbf{C} - \mathbf{H}\|_F^2 \,, \tag{9.8}$$

Instead of using a iterative solver, given an $\mathbf{H}$, we can derive a closed form solution for $\mathbf{C}$, *i.e.*, ,

$$\mathbf{C}^* = (\mathbf{I} + \lambda \mathbf{H}^T\mathbf{H})^{-1}\lambda \mathbf{H}^T\mathbf{H}. \tag{9.9}$$

Plugging the solution of $\mathbf{C}$ back to Eq.(9.8), we arrive at our final union-of-subspaces loss term that only depends on the estimated flow:

$$\begin{aligned}\mathcal{L}_{\text{subspace}} =& \frac{1}{2}\|(\mathbf{I} + \lambda \mathbf{H}^T\mathbf{H})^{-1}\lambda \mathbf{H}^T\mathbf{H}\|_F^2 \\ &+ \frac{\lambda}{2}\|\mathbf{H}(\mathbf{I} + \lambda \mathbf{H}^T\mathbf{H})^{-1}\lambda \mathbf{H}^T\mathbf{H} - \mathbf{H}\|_F^2 \,.\end{aligned} \tag{9.10}$$

Directly applying this loss to the whole image will lead to GPU memory overflow due to the computation of $\mathbf{H}^T\mathbf{H} \in \mathbb{R}^{N\times N}$ (with N the number of pixels in a image). To avoid this, we employ a randomly sampling strategy to sample 2000 flow points in a flow map and compute a loss based on these samples. This strategy is valid because random sampling will not change the intrinsic character of sets.

We remark that this subspace loss requires no prior knowledge of the number of motions in a scene, so it can be used to train a flow network on a *motion-agnostic* dataset. In a single-motion case, it works similarly to the low-rank loss since the optimal loss is closely related to the rank of $\mathbf{H}$ [231]. In a multi-motion case, as long as the epipolar subspaces are disjoint and principle angles between them are below certain thresholds [233], this loss can still serve as a global regularizer. Even when the scene is highly non-rigid or dynamic, unlike the hard epipolar constraint, this loss won't be detrimental to the network training because it will have same values for both ground-truth flows and wrong flows. In Fig. 9.1, we show the results of a typical image pair from KITTI using this constraint, demonstrating the effectiveness of our method.

9.4 Unsupervised Learning of Optical Flow

We formulate our unsupervised optical flow estimation approach as an optimization of image based losses and epipolar constraint losses. In unsupervised optical flow estimation, only photometric loss $\mathcal{L}_{\text{photo}}$ can provide data term. Additionally, we use a smoothness term $\mathcal{L}_{\text{smooth}}$ and our epipolar constraint term $\mathcal{L}_{\mathbf{F}|\text{lowrank}|\text{subspace}}$ as our regularization terms. Our overall loss $\mathcal{L}$ is a linear combination of these three losses

$$\mathcal{L} = \mathcal{L}_{\text{photo}} + \mu_1\mathcal{L}_{\text{smooth}} + \mu_2\mathcal{L}_{\mathbf{F}|\text{lowrank}|\text{subspace}}, \tag{9.11}$$

where μ_1, μ_2 are the weights for each term. We empirically set $\mu_1 = 0.02$ and $\mu_2 = 0.02, 0.01, 0.001$ for $\mathcal{L}_{\mathbf{F}}, \mathcal{L}^*_{\text{lowrank}}, \mathcal{L}_{\text{subspace}}$ respectively.

9.4.1 Image Warping Loss

Similarly to conventional methods, we leverage the most popular brightness constancy assumption, *i.e.*, I^t, I^{t+1} should have similar pixel intensities, colours and gradients. Our photometric error is then defined by the difference between the reference frame and the warped target frame based on flow estimation.

In [116], they target at the case which the illumination may changes from frame to frame and propose a bidirectional census transform $C(\cdot)$ to handle this situation. We adopt this idea to our photometric error. Therefore, our photometric loss is a weighted summation of pixel intensities (or colour) loss $\mathcal{L}_i$, image gradient loss $\mathcal{L}_g$ and bidirectional census loss $\mathcal{L}_c$.

$$\mathcal{L}_{\text{photo}} = \lambda_1 \mathcal{L}_i + \lambda_2 \mathcal{L}_c + \lambda_3 \mathcal{L}_g, \tag{9.12}$$

where $\lambda_1 = 0.5, \lambda_2 = 1, \lambda_3 = 1$ are the weights for each term.

Inspired by [117], we only compute our photometric loss on non-occluded areas O and normalize the loss by the number of pixels of non-occluded regions. We determine a pixel to be occluded or not by forward-backward consistency check. If the sum of its forward and backward flow is above a threshold τ, we set the pixel as occluded. We use $\tau = 3$ in all experiments.

Our photometric loss is thus defined as follows:

$$\mathcal{L}_i = \left[\sum_{i=1}^{N} O_i \cdot \varphi(\hat{I}^t(\mathbf{x}_i) - I^t(\mathbf{x}_i)) \right] / \sum_{i}^{N} O_i \tag{9.13}$$

$$\mathcal{L}_c = \left[\sum_{i=1}^{N} O_i \cdot \varphi(\widehat{C}^t(\mathbf{x}_i) - C^t(\mathbf{x}_i)) \right] / \sum_{i}^{N} O_i \tag{9.14}$$

$$\mathcal{L}_g = \left[\sum_{i=1}^{N} O_i \cdot \varphi(\nabla \hat{I}^t(\mathbf{x}_i) - \nabla I^t(\mathbf{x}_i)) \right] / \sum_{i}^{N} O_i \tag{9.15}$$

where $\hat{I}^t(\mathbf{x}_i) = I^{t+1}(\mathbf{x}_i + \mathbf{v}_i)$ is computed through image warping with the estimated flow, and following [117], we use a robust Charbonnier penalty $\varphi(x) = \sqrt{x^2 + 0.001^2}$ to evaluate differences.

9.4.2 Smoothness Loss

Commonly, there are two kinds of smoothness prior in conventional optical flow estimation: One is piece-wise planar, and the other is piece-wise linear. The first one can be implemented by penalizing the first order derivative of recovered optical flow and the later one is by the second order derivative. For most rigid scenes, piece-wise planar model can provide a better interpolation. But for deformable cases, piece-wise linear model suits better. Therefore, we use a combination of these two models as our smoothness regularization term. We further assumes that the edges in optical flows are edges in reference colour images as well. Formally, our image guided smoothness

term can be defined as:

$$\mathcal{L}_s = \sum \left(e^{-\alpha_1 |\nabla I|} \, |\nabla V| + e^{-\alpha_2 \left|\nabla^2 I\right|} \left|\nabla^2 V\right| \right) / N, \tag{9.16}$$

where $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$ and $V \in \mathbb{R}^{W \times H \times 2}$ is a matrix form of $\mathbf{v}$.

9.5 Experiments

We evaluate our methods on standard optical flow benchmarks, including KITTI [89, 90], MPI-Sintel [68], Flying Chairs [77], and Middlebury [69]. We compare our results with existing optical flow estimation methods based on standard metrics, *i.e.*, endpoint error (EPE) and percentage of optical flow outliers (Fl). We denote our method as EPIFlow.

9.5.1 Implementation details.

Architecture and Parameters. We implemented our EPIFlow network in an end-to-end manner by adopting the architecture of PWC-Net [80] as our base network due to its state-of-the-art performance. The original PWC-Net takes a structure of pyramid and learns on 5 different scales. However, a warping error is ineffective on low resolutions. Therefore, we pick the highest resolution output, upsample it to match the input resolution by bilinear interpolation, and compute our self-supervised learning losses only on that scale. The learning rate for initial training (from scratch) is 10^{-4} and that for fine-tuning is 10^{-5}. Depending on the resolution of input images, the batch size is 4 or 8. We use the same data argumentation scheme as proposed in FlowNet2 [78]. Our network's typical speed varies from 0.07 to 0.25 seconds per frame during the training process, depending on the input image size and the losses used, and is around 0.04 seconds per frame in evaluation. The experiments were tested on a regular computer equipped with a Titan XP GPU. EPIFlow is significantly faster compared with conventional methods.

Pre-training. We pre-trained our network on the Flying Chairs dataset using a weighted combination of the warping loss and smoothness loss. Flying Chairs is a synthetic dataset consisting of rendered chairs superimposed on real-world Flickr images. Training on such a large-scale synthetic dataset allows the network to learn the general concepts of optical flow before handling complicated real-world conditions, *e.g.*, changeable light or motions. To avoid trivial solutions, we disabled the occlusion-aware term at the beginning of the training (*i.e.*, the first two epochs). Otherwise, the network would generate all zero occlusion masks which invalidate losses. The pre-training roughly took forty hours and its returned model was used as an initial model for other datasets.

Table 9.1: **Performance comparison on the KITTI and Sintel optical flow benchmarks.** The metric EPE(noc) indicates the average endpoint error of non-occluded regions while the term EPE(all) is that for all pixels. The KITTI 2015 testing dataset evaluates results by the percentage of flow outliers (Fl). The baseline, gtF, F, low-rank, and sub models were trained on the KITTI VO dataset. The parentheses indicate the corresponding models that were trained on the same data. Note that the current STOA unsupervised method Back2Future Flow [118] uses three frames as input. Best results are marked by bold fonts.

	Method	KITTI 2012				KITTI 2015			Sintel Clean		Sintel Final	
		EPE(all)		EPE(noc)		EPE(all)	EPE(noc)	Fl−all	EPE(all)		EPE(all)	
		train	*test*	*train*	*test*	*train*	*train*	*test*	*train*	*test*	*train*	*test*
nodeep	EpicFlow [234]	(3.09)	3.8	–	1.5	(9.27)	–	26.29%	(2.27)	4.11	(3.56)	6.29
nodeep	MRFlow [88]	–	–	–	–	–	–	12.19%	(1.83)	2.53	(3.59)	5.38
Supervised	SpyNet [79]	9.12	–	–	–	20.56	–	35.07%	(4.12)	6.69	(5.57)	8.43
Supervised	FlowNet2 [78]	4.09	–	–	–	10.06	–	–	2.02	3.96	3.14	6.02
Supervised	PWC-Net [80]	4.14	–	–	–	10.35	–	–	2.55	3.45	3.93	4.60
Supervised	PWC-Net-ft [80]	(1.45)	1.5	–	0.8	(2.16)	–	9.60%	(1.70)	3.86	(2.21)	5.17
Unsupervised	UnsupFlownet [110]	11.30	9.9	4.30	4.6	–	–	–	–	–	–	–
Unsupervised	DSTFlow-ft [111]	(10.43)	12.4	3.29	4.0	16.79	6.96	39.00%	7.10	10.95	7.95	11.80
Unsupervised	DF-Net [225]	3.54	4.4	–	–	8.98	–	25.70%	–	–	–	–
Unsupervised	GeoNet [99]	–	–	–	–	10.81	8.05	–	–	–	–	–
Unsupervised	UnFlow [116]	3.29	–	1.26	–	8.10	–	–	–	9.38	7.91	10.21
Unsupervised	OAFlow [117]	3.55	4.2	–	–	8.88	–	31.20%	7.41	–	7.92	–
Unsupervised	CCFlow [102]	–	–	–	–	–	7.76	–	–	–	–	–
Unsupervised	Back2Future [118]	–	–	–	–	6.59	3.22	22.94%	(3.89)	7.23	(5.52)	8.81
Unsupervised	Our-baseline	3.23	–	1.04	–	7.93	4.21	–	6.72	–	7.31	–
Unsupervised	Our-gtF	2.61	–	1.04	–	6.03	2.89	–	6.15	–	6.71	–
Unsupervised	Our-F	2.56	–	**0.97**	–	6.42	3.09	–	6.21	–	6.73	–
Unsupervised	Our-low-rank	2.63	–	1.07	–	5.91	3.03	–	6.39	–	6.96	–
Unsupervised	Our-sub	2.62	–	1.03	–	6.02	2.98	–	6.15	–	6.83	–
Unsupervised	Our-sub-test-ft	2.61	(**3.2**)	1.03	(**1.1**)	5.56	2.56	(**16.24%**)	3.94	(**6.84**)	5.08	(**8.33**)
Unsupervised	Our-sub-train-ft	(**2.51**)	3.4	(0.99)	1.3	(**5.55**)	(**2.46**)	16.95%	**(3.54)**	7.00	**(4.99)**	8.51

9.5.2 Datasets

KITTI Visual Odometry (VO) Dataset. The KITTI VO dataset contains 22 calibrated sequences with 87,060 consecutive pairs of real-world images. The ground truth poses of the first 11 sequences are available. We fine-tuned our initial model on the KITTI VO dataset using various loss combinations. We chose it for two reasons: (1) it provides ground truth camera poses for every frame, which simplifies the problem of network performance analysis and (2) most scenes in the KITTI VO dataset are stationary and thus can be fitted by an ego-motion. The relative poses (between a pair of images) and camera calibration can be used to compute fundamental matrices. To compare our various methods fairly, we use the first 11 sequences as our training set.

KITTI Optical Flow Dataset. The KITTI optical flow dataset contains two subsets: KITTI 2012 and KITTI 2015, where the first one mostly contains stationary scenes and the latter one includes more dynamic scenes. KITTI 2012 provides 194

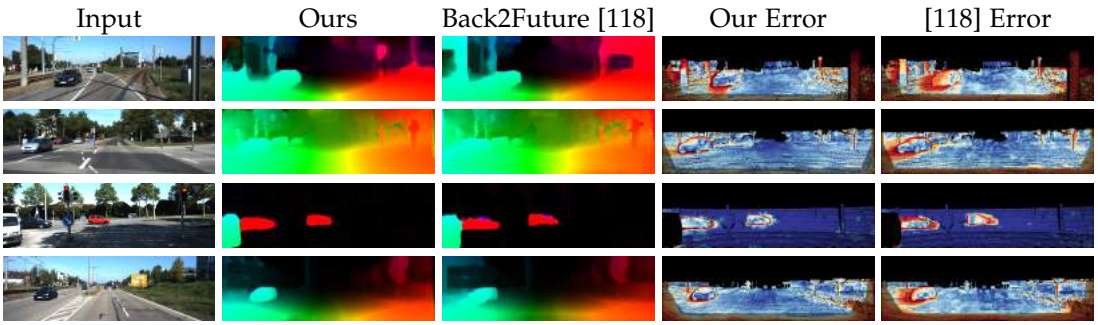


Figure 9.2: **Qualitative results on KITTI 2015 Test dataset.** We compare our method with Back2Future Flow [118]. The second column contains the flows estimated by Our-sub-ft model while the third column contains the results of Back2Future Flow. The flow error visualization is also provided where correct estimates are depicted in blue and wrong ones in red. Consistent with the quantitative analysis, our results are visually better on structural boundaries

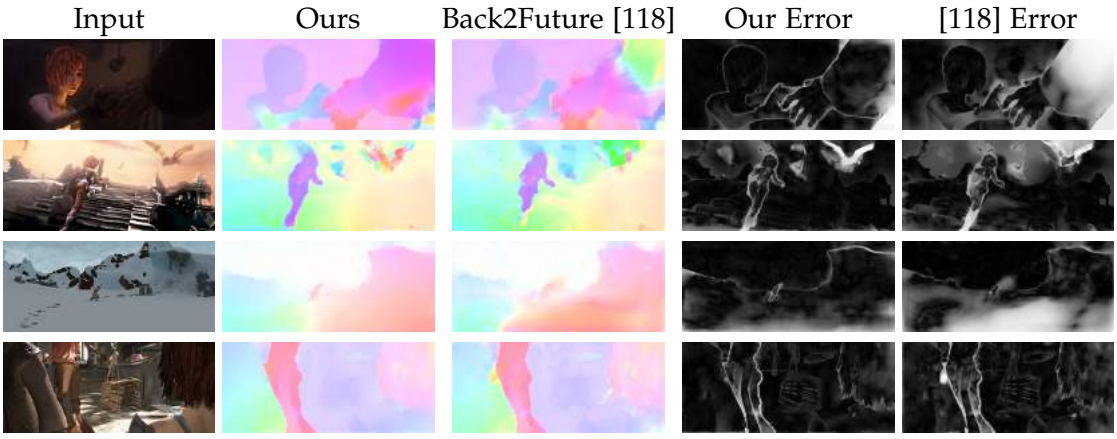


Figure 9.3: **Qualitative results on the MPI Sintel dataset.** This figure shares the same layout with Fig. 9.2 except the top two rows are from the Final set and the two bottom rows are from the Clean set. The errors are visualized in gray on the Sintel benchmark.

annotated image pairs for training and 195 pairs for testing while KITTI 2015 provides 200 pairs for training and 200 pairs for testing. Our training did not use the KITTI datasets' multiple-view extension images.

MPI Sintel Dataset. The MPI Sintel dataset provides naturalistic frames which were captured from an open source movie. It contains 1041 training image pairs with ground truth optical flows and pixel-wise occlusion masks, and also provides 552 image pairs for benchmark testing. The scenes of the MPI Sintel dataset were rendered under two different complexity (Clean and Final). Unlike the KITTI datasets, most scenes in the Sintel dataset are highly dynamic.

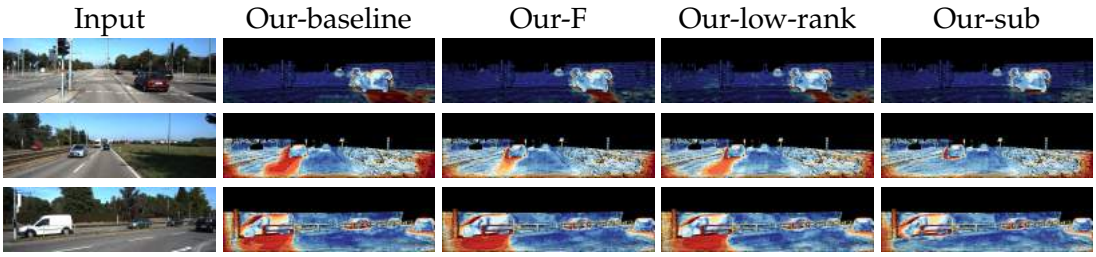


Figure 9.4: **Endpoint error performance of our various models on the KITTI 2015 training dataset.** We compared Our-baseline, Our-F, Our-low-rank, and Our-sub models on the KITTI 2015 dataset to analyze their performance when handling dynamic objects. The results of the Our-sub model are much better.

9.5.3 Quantitative and Qualitative Results

We use the suffix "-baseline" to indicate our baseline model that was trained using only photometric and smoothness loss. "-F" represents the model that was trained using hard fundamental matrix constraint with estimated **F**. "-gtF" means that we used the ground truth fundamental matrix. "-low-rank" refers to the model applying the low rank constraint, and "-sub" is the model using our subspace constraint. "-ft" denotes the model fine-tuned on the datasets.

KITTI VO training results. We report our results that were trained on the KITTI VO dataset in Table 9.1, where our models are compared with various state-of-the-art methods. Our methods outperform all previous learning-based unsupervised optical flow methods with a notable margin. Note that most scenes in KITTI VO dataset are stationary, and therefore the difference between our-gtF, our-F, our-low-rank and our-sub is small across these benchmarks.

Benchmark Fine-tuning Results. We fine-tuned our models on each benchmark and report the results with a suffix '-ft' in Table 9.2. For example, simply following the same hyper-parameters as before, we finetuned our models on the KITTI 2015 testing data. After fine-tuning, Our-sub model shows great performance improvement and achieved an EPE of 2.61 and 5.56 respectively on the KITTI 2012 and KITTI 2015 training datasets, which outperforms all the deep unsupervised methods and many supervised methods. Similarly, on the MPI Sintel trainings dataset, Our-sub-ft model performs best among the unsupervised methods, with an EPE of 3.94 on the Clean images and 5.08 on the Final images. Furthermore, both on the KITTI and Sintel testing benchmarks, our method outperformed the current state-of-the-art unsupervised method Back2Future Flow by a margin. We improve the best unsupervised performance from an Fl of 22.94% to 16.24% on KITTI 2015. The Our-sub-ft model achieved an EPE of 6.84 on the Sintel Clean dataset and 8.33 on the Final set, which are the results that unsupervised methods have never touched before. Additionally, it should be noted that the Back2Future Flow method is based on a multi-frame formulation while our method only requires two frames. Our model is also competitive compared with some fine-tuned supervised networks, such as SpyNet.

Qualitatively, as shown in Fig. 9.2 and Fig. 9.3, compared with the results of Back2Future Flow, the shapes in our estimated flows are more structured and have more explicit boundaries which represent motion discontinuities. This trend is also apparent in the flow error images. For example, on the KITTI 2015 dataset (Fig. 9.2), the results of Back2Future Flow usually bring a larger region of error with crimson colours around the objects.

It should be noted that fine-tuning on the target datasets (*e.g.,* KITTI 2015) does not bring significant improvement because we have trained the models on a real-world dataset KITTI VO. The models have learned the general concepts of realistic optical flows and fine tuning just helps them familiar with the datasets' characteristics. On the KITTI 2012 training set, the fine-tuned model achieves very close results with the Our-sub model, which are respectively 2.61 and 2.62 EPE. Fine-tuning on the Sintel Clean dataset improves the result from 6.15 to 3.94 EPE, because the Sintel Clean dataset renders the synthetic scenes under low complexity and the images are quite different from the real world.

9.5.4 Ablation study

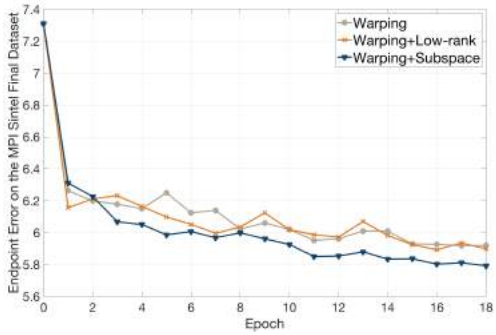


Figure 9.5: **Endpoint error over epochs on the Sintel Final dataset.** We illustrate the endpoint errors over the training epochs when using various combinations of constraints. For all the three methods, the training started from the same pre-trained model 'Our-baseline'. Combing the image warping and subspace constraints outperforms other two methods, which is consistent with the final fine-tuned results reported in Table 9.2.

The Our-F, Our-low-rank and Our-sub models all work well in stationary scenes and they have similar quantitative performance. To further analyze their capabilities in handling general dynamic scenarios, we fine-tuned each method on the KITTI 2015 and Sintel Final dataset. Both of them involve multiple motions in an image while Sintel scenes are more dynamic. As shown in Table 9.2, Our-sub can handle dynamic scenarios best and achieves the lowest EPE in both benchmarks. The hard fundamental constraint shares a similar performance with our baseline model but cannot converge on the Sintal dataset, whose EPE is reported as NaN. It is because a highly dynamic scene does not have a global fundamental $\mathbf{F}$. For the low-rank

Table 9.2: **Fine-tuning results comparison on KITTI 2015 and Sintel Final training sets.** We fine-tuned our models on the training sets of KITTI 2015 and Sintel Final dataset. The term NaN indicates the model cannot converge.

Method	KITTI 2015		Sintel Final
	EPE(all)	EPE(noc)	EPE(all)
Our-baseline-ft	6.16	2.85	5.87
Our-F-ft	6.19	2.85	NaN
Our-low-rank-ft	5.72	2.62	5.59
Our-sub-ft	5.56	2.56	5.08

constraint, its performance is not affected by dynamic objects while it cannot gain information by modeling multiple movements as well. In Fig. 9.5, we provide the validation error curves over the training's early stages on Sintal final dataset. The subspace loss helps the model converge quicker and achieve lower cost than other methods.

9.6 Conclusion

In this chapter, we have proposed effective methods to enforce global epipolar geometry constraints for unsupervised optical flow learning. For a stationary scene, we applied the low-rank constraint to regularize a globally rigid structure. For general dynamic scenes (multi-body or deformable), we proposed to use the union-of-subspaces constraint. Experiments on various benchmarking datasets have proved the efficacy and superiority of our methods compared with state-of-the-art (unsupervised) deep flow methods. In the future, we plan to study the multi-frame extension, *i.e.*, enforcing geometric constraints across multiple frames.

Part VI

Learning depth and camera motion from a monocular video

Chapter 10

Deep Two-View Structure from Motion

Two-view structure-from-motion (SfM) is the cornerstone of 3D reconstruction and visual SLAM. Existing deep learning-based approaches formulate the problem in ways that are fundamentally ill-posed, relying on training data to overcome the inherent difficulties. In contrast, we propose a return to the basics. We revisit the problem of deep two-view SfM by leveraging the well-posedness of the classic pipeline. Our method consists of 1) an optical flow estimation network that predicts dense correspondences between two frames; 2) a normalized pose estimation module that computes relative camera poses from the 2D optical flow correspondences, and 3) a scale-invariant depth estimation network that leverages epipolar geometry to reduce the search space, refine the dense correspondences, and estimate relative depth maps. Extensive experiments show that our method outperforms all state-of-the-art two-view SfM methods by a clear margin on KITTI depth, KITTI VO, MVS, Scenes11, and SUN3D datasets in both relative pose estimation and depth estimation. Note that this framework can be easily adapted to self-supervised manner using the self-supervised losses that proposed in [14, 16].

10.1 Introduction

Two-view structure-from-motion (SfM) is the problem of estimating the camera motion and scene geometry from two image frames of a monocular sequence. As the foundation of both 3D reconstruction and visual simultaneous localization and mapping (vSLAM), this important problem finds its way into a wide range of applications, including autonomous driving, augmented/virtual reality, indoor navigation, and robotics.

Classic approaches to two-view SfM follow a standard pipeline of first matching features/edges between the two images, then inferring motion and geometry from those matches [235, 236, 93, 237, 238, 239, 240]. When imaging conditions are well-behaved (constant lighting, diffuse and rigid surfaces, and non-repeating visual texture), the matching process is well-posed. And, once the matches have been found, the motion and geometry can be recovered.

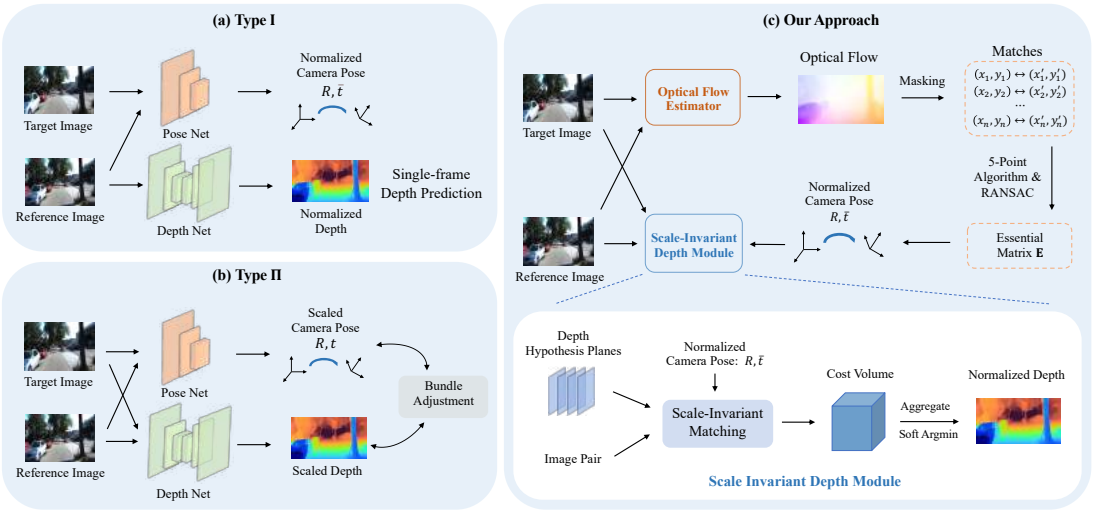


Figure 10.1: **Comparison between our method and previous deep monocular structure-from-motion methods.** We formulate camera pose estimation as a 2D matching problem (optical flow) and depth prediction as a 1D matching problem along an epipolar line. In contrast, previous methods suffer from ill-posedness (either single-frame depth prediction, in the case of Type **I**, or scaled estimates, in the case of Type **II**).

For decades, researchers who work in this area have contented themselves with requiring at least two views, and with recovering only *relative* camera motion and *relative* scene geometry (that is, shape up to an unknown scale factor). Without *a priori* knowledge of scale, or recognizable objects in the scene, it is impossible to recover scene geometry from a single view, nor is it possible mathematically to infer *absolute* scale from two views of a scene.

With the rise of deep learning, a number of researchers have recently explored neural network-based solutions to the problem of two-view SfM. Most of these methods fall into one of two categories. In the first category, which we shall call Type **I**, the problem is treated as a joint optimization task of monocular depth estimation and pose regression [97, 99, 102]. Two networks are used: one to estimate the *up-to-scale* depth from a single image, and another to estimate the *up-to-scale* camera pose from two input images. Both networks act independently in the inference. In the second category, denoted Type **II**, the *scaled* camera pose and the *scaled* depth are inferred from the image pair, and are iteratively refined via multi-view geometry [98, 103, 106].

While the power of deep learning allows both Type **I** and Type **II** solutions to achieve competitive performance on several SfM benchmarks, these methods attempt to solve problems that are fundamentally *ill-posed*, in the sense of Hadamard [241]. The fact that single-view 3D reconstruction is ill-posed is well-known [242, 243, 244], as is the impossibility of recovering absolute scale from two monocular views [92]. To overcome the difficulties of solving these ill-posed problems, deep learning-based approaches rely heavily upon prior knowledge captured in the training data, which in turn constrains their generalizability.

In this chapter, we revisit the use of deep learning for two-view SfM. Our framework follows the classic pipeline in which features are matched between image frames to yield *relative* camera poses, from which *relative* depths are estimated. By combining the strengths of deep learning within a classic pipeline, we are able to avoid ill-posedness, which allows our approach to achieve state-of-the-art results on several benchmarks.

A comparison between our approach and existing pipelines is shown in Fig. 10.1. Our method operates by first estimating dense matching points between two frames using a deep optical flow network [147], from which a set of highly reliable matches are sampled in order to compute the relative camera pose via a GPU-accelerated classic five-point algorithm [245] with RANSAC [96]. Since these relative camera poses have scale ambiguity, the estimated depth suffers from scale ambiguity as well. Therefore, in order to supervise the estimated scale-ambiguous depth with the (scaled) ground truth depth, we propose a novel scale-invariant depth estimation network combined with scale-specific losses to estimate the final relative depth maps. Since the search space of the depth estimation network is reduced to epipolar lines thanks to the camera poses, it yields higher accuracy than directly triangulating the optical flows with the estimated camera poses. We demonstrate the effectiveness of our framework by achieving state-of-the-art accuracy in both pose estimation and depth estimation on KITTI depth, KITTI VO, MVS, Scenes11, and SUN3D datasets.

Our main contributions are summarized as:

1) We revisit the use of deep learning in SfM, propose a new deep two-view SfM framework that avoids ill-posedness. Our framework combines the best of deep learning and classical geometry.
2) We propose a scale-invariant depth estimation module to handle the mismatched scales between ground truth depth and the estimated depth.
3) Our method outperforms all previous methods on various benchmarks for both relative pose estimation and depth estimation under the two-view SfM setting.

10.2 Two-view Geometry: Review and Analysis

The task of two-view SfM refers to estimating the relative camera poses and dense depth maps from two consecutive monocular frames. In classic geometric vision, it is well understood that the camera poses as well as the depth maps can be computed from image matching points alone without any other information [91].[1]

Given a set of image matching points in homogeneous coordinates, $\mathbf{x}_i = \begin{bmatrix} x_i & y_i & 1 \end{bmatrix}^\top$ and $\mathbf{x}'_i = \begin{bmatrix} x'_i & y'_i & 1 \end{bmatrix}^\top$ with known camera intrinsic matrix $\mathbf{K}$, the two-view SfM task is to find a camera rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ as well as the corresponding 3D homogeneous points $\mathbf{X}_i$ and $\mathbf{X}'_i$ such that:

$$\mathbf{x}_i = \mathbf{K} \begin{bmatrix} \mathbf{I} \mid \mathbf{0} \end{bmatrix} \mathbf{X}_i \quad \mathbf{x}'_i = \mathbf{K} \begin{bmatrix} \mathbf{R} \mid \mathbf{t} \end{bmatrix} \mathbf{X}'_i \qquad \forall i. \tag{10.1}$$

[1] Excluding degenerate cases.

A classical method to solve this problem consists of three consecutive steps: 1) Computing the essential matrix $\mathbf{E}$ from the image matching points $\mathbf{x}_i$ and $\mathbf{x}'_i$; 2) Extracting the relative camera pose $\mathbf{R}$ and $\mathbf{t}$ from the essential matrix $\mathbf{E}$; 3) Triangulating the matching points $\mathbf{x}_i$ and $\mathbf{x}'_i$ with the camera pose to get 3D points $\mathbf{X}_i$ and $\mathbf{X}'_i$.

All steps in this pipeline are *well-posed* problems. The essential matrix $\mathbf{E}$ can be solved with at least 5 matching points using the equation below:

$$\mathbf{x}_i'^{\top}\mathbf{K}^{-\top}\mathbf{E}\mathbf{K}^{-1}\mathbf{x}_i = 0 \qquad \forall i. \tag{10.2}$$

$\mathbf{R}$ and $\mathbf{t}$ can be computed from $\mathbf{E}$ using matrix decomposition such that $\mathbf{E} = \mathbf{S}\mathbf{R}$, where $\mathbf{S}$ is a skew symmetric matrix and $\mathbf{R}$ is a rotation matrix. Since for any non-zero scaling factor α, $\left[\alpha\mathbf{t}\right]_\times \mathbf{R} = \alpha \left[\mathbf{t}\right]_\times \mathbf{R} = \alpha\mathbf{E}$ provides a valid solution, there is a scale ambiguity for relative camera pose estimation. The 3D points $\mathbf{X}_i$ and $\mathbf{X}'_i$ can be computed by triangulation with a global scale ambiguity.

The above method assumes the ideal case in which all image points are perfectly matched. To handle mismatched points in real scenarios, researchers have established a classical standard pipeline to estimate geometry information from two consecutive frames [92].

10.2.1 The Classic Standard Pipeline

With decades of development and refinement, the classic standard pipeline [92] is widely used in many conventional state-of-the-art SfM and vSLAM systems [93, 94, 95]. Since all geometry information can be recovered from image matching points, the key is to recover a set of (sparse or dense) accurate matching points. To this end, the pipeline often starts with sparse (or semi-dense) distinct feature extraction and matching to get sparse matching points, as sparse / semi-dense matching is more accurate than dense matching. To further refine the matching results, the RANSAC scheme [96] is used to filter the matching points that do not fit the majority motion. These outliers often include mismatches and dynamic objects in a scene. After retrieving the camera poses from the refined matching points, the depth of these points can be computed via triangulation. In some cases, if it is desired to estimate dense depth maps rather than the sparse 3D points, multi-view stereo matching algorithms can be used to recover the dense depth maps with the estimated camera poses.

The Achilles' heel of this pipeline is therefore the matching of points. Conventional matching algorithms often suffer low accuracy on non-Lambertian, blurry, and textureless surfaces. However, this shortage can be largely alleviated by deep learning [80, 246, 147]. With sufficient training data, such networks can learn to handle these scenarios. In our proposed approach, we leverage a deep optical flow network [147] to compute these correspondences.

10.2.2 Deep Learning based Methods

As discussed earlier, two-view SfM requires to estimate both camera poses and dense depth maps. Existing deep learning based methods either formulate the problem as

pose regression and monocular depth estimation (Type **I**) or as pose regression and multi-view stereo matching (Type **II**). We analyze both types of methods below.

Type I methods consist of a monocular depth estimation network and a pose regression network. The two-view geometry constraints are used as self-supervisory signals to regularize both camera poses and depth maps [97, 99, 100, 101, 102]. As a result, most of these approaches are self-supervised. Because single-view depth estimation is inherently ill-posed, as discussed earlier, these methods are fundamentally limited by how well they can solve that challenging problem.

Since the two-view geometry constraints are only suitable for a stationary scene, SfMLearner [97] simultaneously estimates an explainability mask to exclude the dynamic objects while GeoNet [99] utilizes an optical flow module to mask out these outliers by comparing the rigid flow (computed by camera poses and depth maps) with the non-rigid flow (computed by the optical flow module). Other methods focus on implementing more robust loss functions such as ICP loss [100], motion segmentation loss [102], or epipolar loss [101].

Type II methods require two image frames to estimate depth maps and camera poses at test time (unlike Type I methods, which estimate depth from a single frame). Most supervised deep methods fall into this category. As a pioneer of this type, DeMoN [98] concatenates a pair of frames and uses multiple stacked encoder-decoder networks to regress camera poses and depth maps, implicitly utilizing multi-view geometry.

Similar strategies have been adapted by [103, 104, 105, 106] by replacing generic layers between camera poses and depth maps with optimization layers that explicitly enforce multi-view geometry constraints. For example, BANet [103] parameterizes dense depth maps with a set of depth bases and imposes bundle adjustment as a differentiable layer into the network architecture. Wang *et al.* [105] use regressed camera poses to constrain the search space of optical flow, estimating dense depth maps via triangulation. DeepV2D [106] separates the camera pose and depth estimation, iteratively updating them by minimizing geometric reprojection errors. Similarly, DeepSFM [107] initiates its pose estimation from DeMoN [98], sampling nearby pose hypotheses to bundle adjust both poses and depth estimation. Nevertheless, with the ground truth depth as supervision, it requires the pose regression module to estimate camera poses with absolute scale, which is generally impossible from a pair or a sequence of monocular frames alone [92]. To mitigate this ill-posed problem, they utilize dataset priors and semantic knowledge of the scene to estimate the absolute scale.

10.3 Method

In this section, we propose a new deep two-view structure-from-motion framework that aims to address, via deep learning, the Achilles' heel of the classical SfM pipeline (*viz.*, matching). Our method is able to find better matching points and therefore more accurate depth maps, especially for textureless and occluded areas. At the

same time, it follows the wisdom of classic methods by avoiding ill-posed problems. By combining the best of both worlds, our approach is able to achieve state-of-the-art results, outperforming all previous methods by a clear margin.

Following the classic standard pipeline [92], we formulate the two-frame structure-from-motion problem as a three-step process: 1) match corresponding points between the frames, 2) estimate the essential matrix, and hence the relative camera pose, 3) estimate dense depth maps, up to an unknown scale factor. These steps, along with the loss function used for training, are described in more detail in the following subsections.

10.3.1 Optical Flow Estimation

A fundamental problem in computer vision, optical flow estimation has been extensively studied for several decades [66]. With the recent progress in deep learning, deep optical flow methods now dominate various benchmarks [247, 68] and can handle large displacements as well as textureless, occluded, and non-Lambertian surfaces. In our framework, we utilize the state-of-the-art network, DICL-Flow [147], to generate dense matching points between two consecutive frames. This method uses a displacement-invariant matching cost learning strategy and a soft-argmin projection layer to ensure that the network learns dense matching points rather than image-flow regression. The network was pre-trained on synthetic datasets (FlyingChairs [77] and FlyingThings [248]) to avoid data leakage, *i.e.,* , the network was not pre-trained on any of the test datasets.

10.3.2 Essential Matrix Estimation

The traditional approach to estimating camera pose between two image frames is to match sparse points, *e.g.,* , SIFT features [249], Then, given a set of matching points $\mathbf{x} \leftrightarrow \mathbf{x}'$ and the camera intrinsic matrix $\mathbf{K}$, the essential matrix [91] $\mathbf{E}$ can be recovered from the five-point algorithm [250, 245]. By decomposing the essential matrix as $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$, the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$ can be recovered up to a scale ambiguity. Conventionally, outliers in the matching points are filtered using robust fitting techniques such as RANSAC [96]. RANSAC repeatedly estimates the essential matrix from randomly sampled minimal matching sets and selects the solution that is satisfied by the largest proportion of matching points under a certain criterion.

Unlike all previous deep learning-based methods that regress the camera poses from input images, we use matching points to compute the camera poses. The key question is this: How to robustly filter the noisy dense matches from optical flow in order to retain only the high quality matches? There are multiple ways to filter out unreliable matching points such as flow uncertainty, consistency check, or using a network to regress a mask. Empirically, we find that simply using SIFT keypoint locations (note that we do *not* use SIFT matching) to generate a mask works well in all datasets. The hypothesis is that optical flow is more accurate in rich textured areas.

The optical flow matches at the locations within the mask are filtered by RANSAC with GPU acceleration, to avoid distraction by dynamic objects. After retrieving the essential matrix $\mathbf{E}$, the camera pose $(\mathbf{R}, \mathbf{t})$ is recovered using matrix decomposition.

10.3.3 Scale-Invariant Depth Estimation

Once we have recovered the up-to-scale relative camera pose, with the dense matching points from optical flow estimation, we could compute the dense depth map by performing triangulation. However, such an approach would not take advantage of the epipolar constraint. As a result, we perform the matching again by constraining the search space to epipolar lines computed from the relative camera pose. This process is similar to multi-view stereo (MVS) matching with one important difference: we do not have the absolute scale in inference. With the up-to-scale relative pose, if we were to directly supervise the depth estimation network with ground truth depth, there would be a mismatch between the scale of the camera motion and the scale of the depth map.

Previous approaches. To resolve this paradox, previous methods either use a scale-invariant loss [65] or regress the absolute scale with a deep network [98, 106, 107]. The scale-invariant loss ℓ_{SI} is defined as:

$$\ell_{\text{SI}} = \sum_{\mathbf{x}} \Big(\log(d_{\mathbf{x}}) - \log(\hat{d}_{\mathbf{x}}) + \eta(d, \hat{d}) \Big)^2, \tag{10.3}$$

where $d_{\mathbf{x}}$ and $\hat{d}_{\mathbf{x}}$ are the ground truth and estimated depth, respectively, at pixel $\mathbf{x}$; and $\eta(d, \hat{d}) = \frac{1}{N} \sum_{\mathbf{x}} \Big(\log(\hat{d}_{\mathbf{x}}) - \log(d_{\mathbf{x}}) \Big)$ measures the mean log difference between the two depth maps. While working for the direct depth regression pipelines, the scale-invariant loss introduces an ambiguity for network learning as the network could output depth maps with different scales for each sample. This may hinder the principle of plane-sweep, where depth maps with consistent scale are desired. Plane sweep [251] is the process that enforces the epipolar constraint, which reduces the search space from 2D to 1D.

Plane-sweep powered networks require consistent scale during the training and testing process. For example, if we train a network with absolute scales and test it with a normalized scale, its performance will drop significantly (we provide an ablation study in Section 10.4.5). Since it is impossible to recover the absolute scale from two images, previous methods [98, 106, 107] use a network to regress a scale to mimic the absolute scale in inference. This strategy slightly alleviates the scale paradox at the cost of making the problem ill-posed again.

Scale-Invariant Matching. To solve this paradox and keep the problem well-posed, we propose a scale-invariant matching process to recover the up-to-scale dense depth map. Mathematically, given an image point $\mathbf{x}$, we generate L matching candidates $\{\mathbf{x}'_l\}_{l=1}^{L}$:

$$\mathbf{x}'_l \sim \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} (\mathbf{K}^{-1}\mathbf{x}) d_l \\ 1 \end{bmatrix}, \tag{10.4}$$

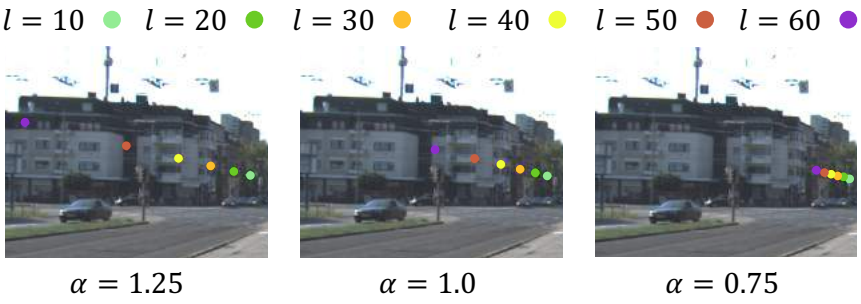


Figure 10.2: **The Effect of Various Scale Factors during Plane Sweep.** For a certain pixel, we visualize its six depth hypotheses with different colors in the target frame. As the scale factor α changes, the sampling distribution varies.

where $d_l = (L \times d_{\text{min}})/l,\ (l = 1, ..., L)$ is the depth hypothesis and d_{min} is a fixed minimum depth.

In the standard plane-sweep MVS setting, the sampling distribution of matching candidates varies depending on the ground truth scale. We illustrate the changes of sampling distribution along with the change of scale $\alpha = \|\mathbf{t}\|_2$ in Fig. 10.2. Since we do not know the absolute scale in our problem, we normalize the translation vectors to $\bar{\mathbf{t}} \equiv \mathbf{t}/\alpha$ such that $\|\bar{\mathbf{t}}\|_2 = 1$. As shown in Eq. (10.4), with fixed $\{d_l\}_{l=1}^{L}$ and normalized translation $\bar{\mathbf{t}}$, the distribution of matching candidates $\{\mathbf{x}'_l\}_{l=1}^{L}$ are now invariant to scale.

To make the estimated and ground truth depths compatible, according to Eq. (10.4), we need to scale the estimated depth $\hat{\mathbf{d}}$ correspondingly to match the ground truth depth $\mathbf{d}$:

$$\mathbf{d} \sim \alpha_{\text{gt}} \hat{\mathbf{d}}, \tag{10.5}$$

where α_{gt} refers to the ground truth scale.

This scale-invariant matching strategy plays a crucial role in our framework as it makes our network no longer suffer from the scale misalignment problem. As shown in Table 10.9, with the scale-invariant matching, our network achieves similar performance comparing with the theoretical upper bound, *i.e.*, , the Oracle model that is trained and tested with ground truth poses.

10.3.4 Loss Function

Our framework is trained in an end-to-end manner with the supervision of ground truth depth maps and ground truth scales. Given a predicted depth $\hat{\mathbf{d}}$ and a ground truth depth $\mathbf{d}$, we supervise the depth using the smooth ℓ_1 loss:

$$\mathcal{L}_{\text{depth}} = \sum_{\mathbf{x}} \ell_{\text{smooth}} \left(\alpha_{\text{gt}} \hat{\mathbf{d}}_{\mathbf{x}} - \mathbf{d}_{\mathbf{x}} \right), \tag{10.6}$$

where $\ell_{\text{smooth}}(z) = 0.5z^2$ if $|z| < 1, |z - 0.5|$ otherwise. It should be noted that our predicted depth is up-to-scale and does not require ground truth scale at inference time.

If both ground truth camera poses $(\mathbf{R}, \mathbf{t})$ and ground truth depth d_{gt} are given, we can also update the optical flow network by computing the rigid flow $\mathbf{u}_\mathbf{x} \equiv \mathbf{x}' - \mathbf{x}$ for 2D point $\mathbf{x}$:

$$\mathbf{x}' \sim \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} (\mathbf{K}^{-1}\mathbf{x}) d_{\text{gt}} \\ 1 \end{bmatrix}. \tag{10.7}$$

The rigid flow can work as a supervision signal, computing the ℓ_2 distance with the estimated optical flow $\hat{\mathbf{u}}_\mathbf{x}$:

$$\mathcal{L}_{\text{flow}} = \sum_{\mathbf{x}} \left(\hat{\mathbf{u}}_\mathbf{x} - \mathbf{u}_\mathbf{x}\right)^2. \tag{10.8}$$

The total loss function of our framework is then given by:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{depth}} + \lambda \mathcal{L}_{\text{flow}}, \tag{10.9}$$

where λ is a trade-off parameter. We set $\lambda = 1$ if fine-tuning the optical flow estimator while make $\lambda = 0$ if just using the flow model pretrained on synthetic datasets.

10.4 Experiments

In this section, we provide quantitative and qualitative results of our framework on various datasets, showing comparison with state-of-the-art SfM methods. We also provide an extensive ablation study to justify our framework design. Due to the scale-ambiguity nature of the two-view SfM problem, we scale the results of ours and others using the same scaling strategy as in [106, 103]. For all experiments, our optical flow estimator is [147] while the architecture of depth estimator is based on [251].

10.4.1 Network and Hyper-parameter Selection

Our framework consists of two matching modules: an optical flow module and a depth estimation module. We select the current off-the-shelf state-of-the-art network, DICL-Flow [147], as our optical flow module and the DPSNet [251] with our proposed scale-invariant modification as our depth estimation module.

We use the SIFT keypoint locations to mask the optical flow before feeding into the RANSAC. For the sake of completeness, we also provide the results of using different keypoint detectors in Table 10.1, including SURF [252] and FAST [253]. While all of them achieve state-of-the-art performance in all three datasets, the SIFT keypoints have the overall best performance.

For essential matrix estimation, we use 512 threads on each GPU to compute essential matrix hypotheses. Each GPU thread randomly selects 5 points from the

Table 10.1: **Various Keypoint Detection Methods for Optical Flow Masking.** We keep the predicted optical flow the same while using the keypoint locations of FAST, SURF, and SIFT to mask the flow correspondences.

Model	MVS		Scenes11		Sun3D	
	Rot	Tran	Rot	Tran	Rot	Tran
DeepSFM [107]	2.824	9.881	0.403	5.828	1.704	13.107
Our-FAST	1.832	3.849	0.327	**1.492**	1.489	11.430
Our-SURF	**1.744**	**3.643**	0.372	1.501	1.587	12.226
Our-SIFT	2.417	3.878	**0.276**	2.041	**1.391**	**10.757**

masked matching points and estimates an essential matrix hypothesis using the 5-point algorithm [250]. We choose the hypothesis with the most inliers using the RANSAC scheme. We empirically set the RANSAC inlier error threshold $\tau = 0.0001$ and set a maximum iteration $\theta = 20$ for MVS, Scenes11 and SUN3D datasets, and $\theta = 5$ for the KITTI dataset. For the depth estimation module, we set the number of matching candidates $L = 96$ for KITTI, MVS, Scenes11, and SUN3D datasets. We use a normalized minimum depth $d_{\text{min}} = 1.0$ for the KITTI dataset and $d_{\text{min}} = 0.5$ for MVS, Scenes11, and SUN3D datasets.

We implement our framework in PyTorch with Automatic Mixed Precision. For KITTI dataset, we use a crop size of $[256, 768]$ and a batch size of 32, and train the network for 10 epochs. The initial learning rate is set to 0.0005 and dropped by half at 3 and 8 epochs. The total training time is 40 hours on eight NVIDIA Tesla V100 GPUs. For the MVS, Scenes11 and SUN3D datasets, we leverage the same training protocol as provided in [107]. The crop size is set to $[256, 384]$ and the batch size is set to 64. We jointly train our network on all three datasets for 10 epochs and use the same initial learning rate of 0.0005 and drop by half at the fifth epoch. The total training time for these three datasets is 85 hours. Our data augmentation includes random flipping, color jittering, resizing, and cropping.

10.4.2 Datasets

KITTI Depth [247] is primarily designed for depth evaluation in autonomous driving scenarios, which does not take camera motions and dynamic objects into account.. The Eigen split [65], which contains 697 single frames for testing, is a widely used split for evaluating single image depth estimation. For the SfM task, the close to static camera motions and dynamic objects will lead to ill-posed situations. To better evaluate the performance of SfM algorithms on Eigen Split, we build the Eigen SfM Split that mostly satisfies two-view SfM assumptions. Specifically, we first pair each frame with its next frame then manually remove these pairs with small relative translations (less than 0.5 meters) or contain large dynamic objects[2]. We use the remaining 256 frames to construct our Eigen SfM Split.

KITTI VO [247] is primarily used for evaluating camera pose estimation. It con-

[2]We define a dynamic object which occupies more than 20% pixels of a scene as a large dynamic object.

tains ten sequences (more than 20k frames) with ground truth camera poses. According to the setting of [97], we test our pose estimation accuracy on the "09" and "10" sequences, using left camera images in a set of rolling two-frame pairs.

MVS, Scenes11, and SUN3D. MVS is collected from several outdoor datasets by [98]. Different from KITTI which is built through video sequences with close scenes, MVS has outdoor scenes from various sources. Scenes11 [98] is a synthetic dataset generated by random shapes and motions. It is therefore annotated with perfect depth and pose, though the images are not realistic. SUN3D [254] provides indoor images with noisy depth and pose annotation. We use the SUN3D dataset post-processed by [98], which discards the samples with a high photo-consistency error.

10.4.3 Depth Evaluation

We perform depth evaluation on KITTI Depth, KITTI VO, MVS, Scenes11, and SUN3D datasets.

KITTI Depth. We compare our framework with both types of deep SfM methods using seven commonly used depth metrics [65]. We also leverage one disparity metric D1-all[3] as it measures the precision of the depth estimation. Since the Type **I** are unsupervised methods and they all perform single frame depth estimation in inference, we also compare with state-of-the-art supervised single image depth estimation methods such as DORN [255] and VNL [256] as they can be viewed as the upper bounds of Type **I** methods.

Quantitative results are shown in Table 10.2. Although only using a flow estimator pre-trained on synthetic datasets, our method beats all previous methods with a clear margin on various metrics, *e.g.*, , 2.273 against 2.727 in RMSE. Especially, our method largely outperforms DeepV2D although DeepV2D used ground truth camera pose and five-frame sequences for training. Note that there is a number of frames in the Eigen split that do not strictly satisfy the rigid SfM assumption such as stationary scene and notable camera motions. When only keeping the frames that satisfy SfM assumptions, *i.e.*, , on the Eigen SfM split, our method achieves even better accuracy, with 3.1% *vs* 9.1% in D1-all. Fig. 10.3 illustrates some qualitative results with a comparison to state-of-the-art single image method [255] and deep SfM method [106].

KITTI VO. To compare the depth estimation results on KITTI VO dataset, we use the state-of-the-art monocular depth estimation network Monodepth2 [257] as our main competitor. For a fair comparison, we re-train both our method and the Monodepth2 [257] with ground truth depth supervision on the first 9 sequences (Seq 00 to Seq 08) of the KITTI VO dataset. We add a suffix "-sup" to Monodepth2 to denote that the model is trained with ground truth depth. We report their depth estimation accuracy on the 9^{th} and 10^{th} sequences. As shown in Table 10.3, our method reduces the error rate of D1_all metric for more than 73%, comparing with

[3]Percentage of stereo disparity outliers. We convert the estimated depth to disparities using the focal length and baseline provided by KITTI.

Table 10.2: **Depth Evaluation on KITTI Depth Dataset.** We compare our results to state-of-the-art single-frame depth estimation methods and deep SfM methods on the KITTI depth Eigen split. We evaluate all SfM methods under two-view SfM setting for a fair comparison. The "Eigen SfM" split (256 frames) excludes frames that are close to static or contain many dynamic objects in the Eigen split. The type **S** means supervised single frame depth estimation. Note that Type **I** methods are unsupervised methods. Bold indicates the best.

Split	Type	Method	lower is better					higher is better		
			Abs Rel	Sq Rel	RMSE	RMSE_{log}	D1-all	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen	**S**	DORN [255]	0.072	0.307	2.727	0.120	0.163	0.932	0.984	0.994
		VNL [256]	0.072	-	3.258	0.117	0.176	0.938	**0.990**	**0.998**
	I	SfMLearner [97]	0.208	1.768	6.856	0.283	-	0.678	0.885	0.957
		GeoNet [99]	0.155	1.296	5.857	0.233	-	0.793	0.931	0.973
		DFNet [225]	0.150	1.124	5.507	0.223	-	0.806	0.933	0.973
		CCNet [102]	0.140	1.070	5.326	0.217	-	0.826	0.941	0.975
		GLNet [101]	0.099	0.796	4.743	0.186	-	0.884	0.955	0.979
	II	BANet [103]	0.083	-	3.640	0.134	-	-	-	-
		DeepV2D [106]	0.064	0.350	2.946	0.120	0.142	0.946	0.982	0.991
		Ours	**0.055**	**0.224**	**2.273**	**0.091**	**0.107**	**0.956**	0.984	0.993
Eigen SfM	**S**	DORN [255]	0.067	0.295	2.929	0.108	0.130	0.949	0.988	0.995
		VNL [256]	0.065	0.297	3.172	0.106	0.168	0.945	0.989	0.997
	II	DeepV2D[106]	0.050	0.212	2.483	0.089	0.091	0.973	0.992	0.997
		Ours	**0.034**	**0.103**	**1.919**	**0.057**	**0.031**	**0.989**	**0.998**	**0.999**

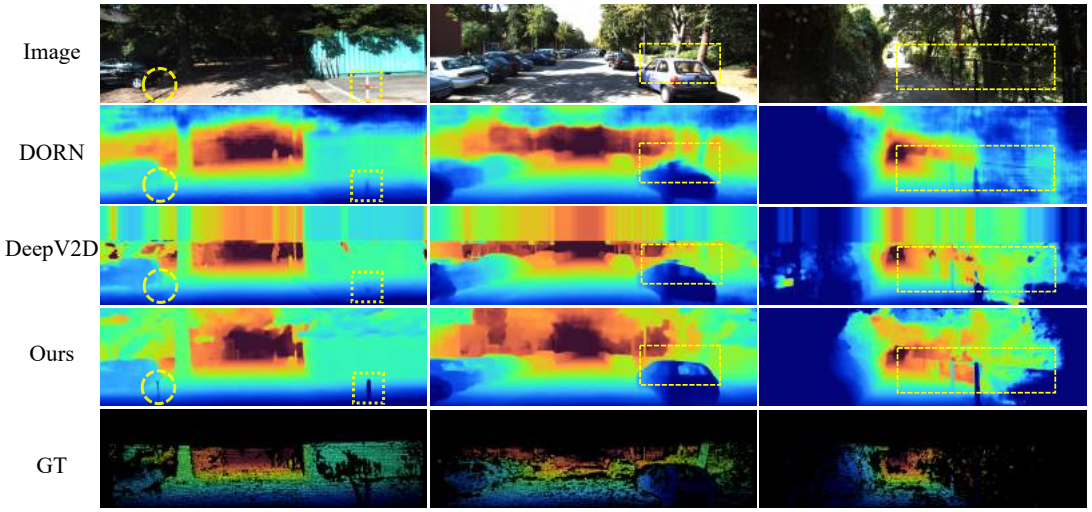
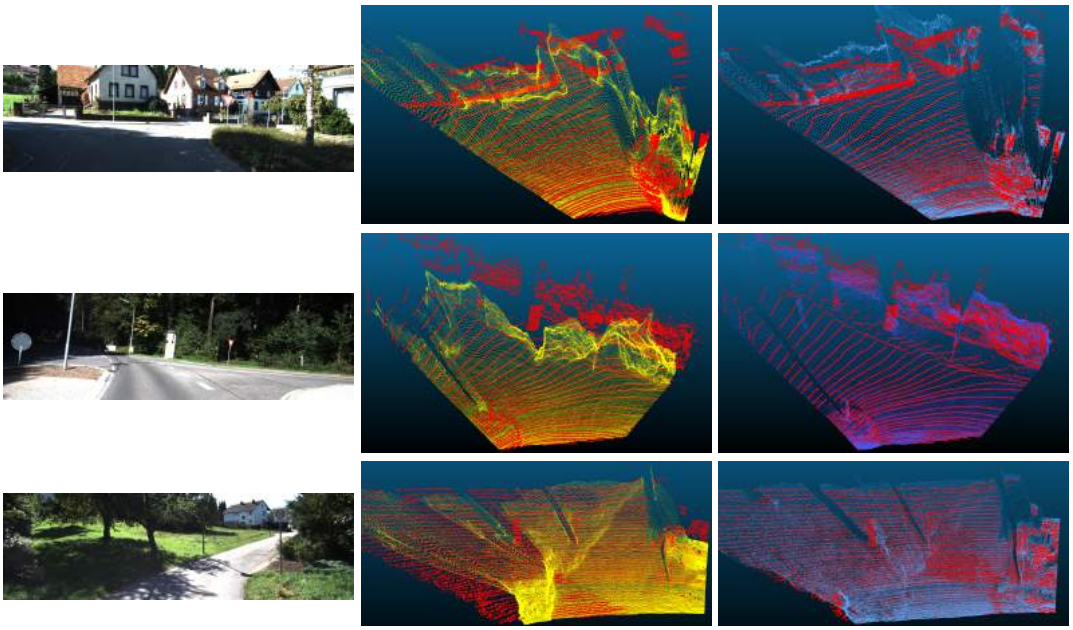


Figure 10.3: **Qualitative Results on the KITTI Dataset.** The yellow circles and boxes in the top row highlight tiny poles which are captured more accurately by our method.

Table 10.3: **Quantitative Results on KITTI VO dataset.** The monodepth2-sup (Mono-sup) [257] model was trained with ground truth depth maps.

Method	Abs Err (m)	Abs Rel	Sq Rel	RMSE (m)	RMSE_{log}	D1-all	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Mono-sup	1.7958	0.0935	0.4842	3.6140	0.1478	26.1221	0.8933	0.9751	0.9938
Ours	**0.9294**	**0.0442**	**0.1618**	**2.2164**	**0.0789**	**7.0159**	**0.9766**	**0.9948**	**0.9981**



Reference Image Monodepth2-sup [257] Ours

Figure 10.4: **Point Cloud Comparison on KITTI VO.** Red: LiDAR; Yellow: Monodepth2 [257] with ground truth supervision; Blue: Ours. Our point clouds are more aligned with the ground truth LiDAR points while the Monodepth2-sup often wrongly estimate depth for grass and trees.

the Monodepth2-sup [257]. We provide a point cloud comparison in Fig. 10.4, where our point clouds are more aligned with the ground truth LiDAR points while the Monodepth2-sup often wrongly estimate depth for grass and trees.

MVS, Scenes11, and SUN3D. We compare our framework to state-of-the-art Type **II** methods under two-view SfM setting using metrics by [98]. We use the same depth inference strategy as in [107]. As shown in Table 10.4, our method achieves superior performance on all metrics among all three datasets comparing with the previous state-of-the-art Type **II** methods. Fig. 10.5 illustrates some qualitative results.

10.4.4 Camera Pose Estimation

We compare the camera pose estimation accuracy with Type **I** and Type **II** SfM methods on KITTI VO, MVS, Scenes11, and SUN3D datasets.

KITTI VO. We compare our visual odometry results on the 9^{th} and 10^{th} sequences of KITTI VO dataset with the state-of-the-art SfM methods in Table 10.5 using common visual odometry evaluation criteria. Since other competitors are fine-tuned on

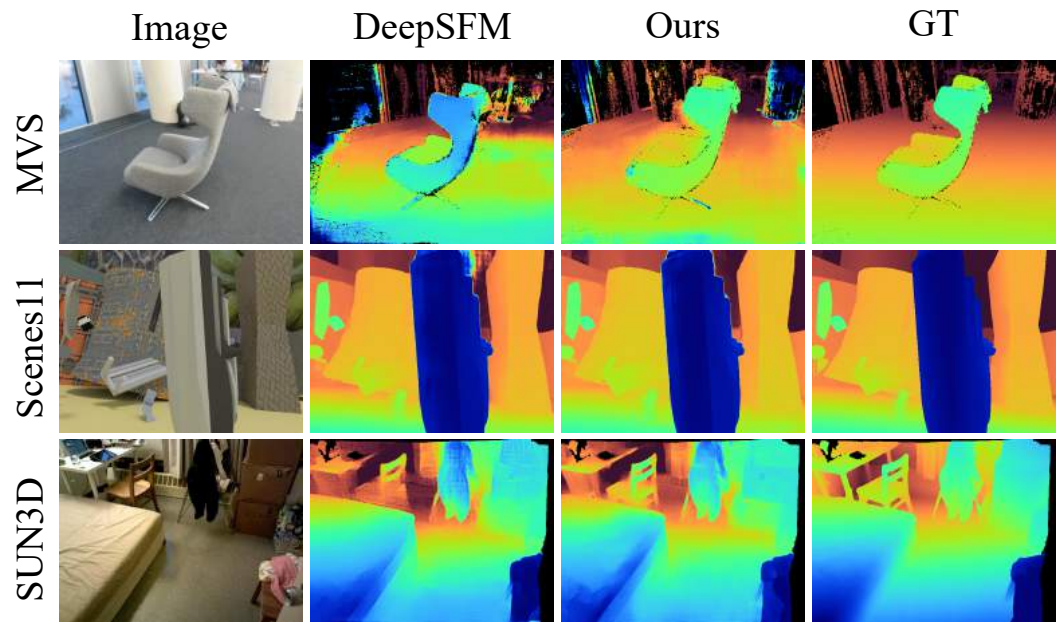


Figure 10.5: **Qualitative Examples on MVS, Scenes11, and SUN3D Datasets,** where our method consistently achieves better results.

Table 10.4: **Depth Estimation Results on MVS, Scenes11, and SUN3D Datasets**. The results of baseline SIFT come from [98].

Method	MVS Dataset			Scenes11 Dataset			Sun3D Dataset		
	Depth			Depth			Depth		
	L1-inv	Sc-inv	L1-rel	L1-inv	Sc-inv	L1-rel	L1-inv	Sc-inv	L1-rel
Base-SIFT	0.056	0.309	0.361	0.051	0.900	1.027	0.029	0.290	0.286
COLMAP [94]	-	-	0.384	-	-	0.625	-	-	0.623
DeMoN [98]	0.047	0.202	0.305	0.019	0.315	0.248	0.019	0.114	0.172
LS-Net [104]	0.051	0.221	0.311	0.010	0.410	0.210	0.015	0.189	0.650
BANet [103]	0.030	0.150	0.080	0.080	0.210	0.130	0.015	0.110	0.060
DeepSFM [107]	0.021	0.129	0.079	0.007	0.112	0.064	0.013	0.093	0.072
Ours	**0.015**	**0.102**	**0.068**	**0.005**	**0.097**	**0.058**	**0.010**	**0.081**	**0.057**

Table 10.5: **Full Sequence Visual Odometry on KITTI VO.** Note that our network is trained on synthetic datasets and compute camera poses from only two consecutive frames while other methods are fine-tuned on the KITTI VO dataset and take multiple frames to estimate the camera poses. Bold indicates the best.

Method	Seq 09					Seq 10				
	t_{err}	r_{err}	ATE	RPE (m)	RPE (°)	t_{err}	r_{err}	ATE	RPE (m)	RPE (°)
SfMLearner [97]	8.28	3.07	24.31	0.099	0.140	12.20	2.96	20.87	0.120	0.154
SC-SfMLearner [258]	7.64	2.19	15.02	0.095	0.102	10.74	4.58	20.19	0.105	0.107
CC [102]	6.92	1.77	29.00	0.095	0.088	7.97	3.11	13.77	0.097	0.116
DF-VO [259]	2.47	**0.30**	11.02	0.055	0.037	1.96	**0.31**	3.37	0.047	0.042
Ours	**1.70**	0.48	**6.87**	**0.016**	**0.034**	**1.49**	0.55	**2.26**	**0.010**	**0.040**

the KITTI VO dataset in an unsupervised manner, for a fair comparison we do not train the optical flow estimator used for computing the camera poses on KITTI, but rather use the network pre-trained on synthetic datasets, without further training. We report the average translational error as $t_{\text{err}}(\%)$ and rotational error as $r_{\text{err}}(^\circ/100m)$. We also adopt the metric absolute trajectory error (ATE) on full sequence and relative pose error (RPE) in meters and degrees. For all results, we align the predicted trajectories to the ground truth via least square optimization [260]. Our approach performs better than all other methods over these five metrics. In Fig. 10.6 we visualize the full sequence odometry trajectories of the 9^{th} and 10^{th} sequences. Our results are more aligned with the ground truth trajectories. It is worth noting that our model are only trained on sythetic datasets while the other methods are fine-tuned on the KITTI VO dataset and take more frames to estimate the camera poses.

MVS, Scenes11 and SUN3D. The competitors use ground truth poses to train their pose estimation module on these three datasets, while we use the ground truth poses to fine-tune our optical flow model using Eq. (10.8). As shown in Table 10.6, our method beats the previous state-of-the-art on all three datasets with a clear margin, *e.g.*, , 60.8% better in translation estimation on MVS dataset and 31.5% better in rotation estimation on Scenes11 dataset. We also verify the effectiveness of rigid flow supervision (Eq (10.8)) in Table 10.7. With fine-tuning, the translation errors are largely suppressed, and the rotation errors are notably reduced. It is worth noting that our model that was pretrained on synthetic datasets has already achieved comparable performance than previous state-of-the-art deep SfM methods.

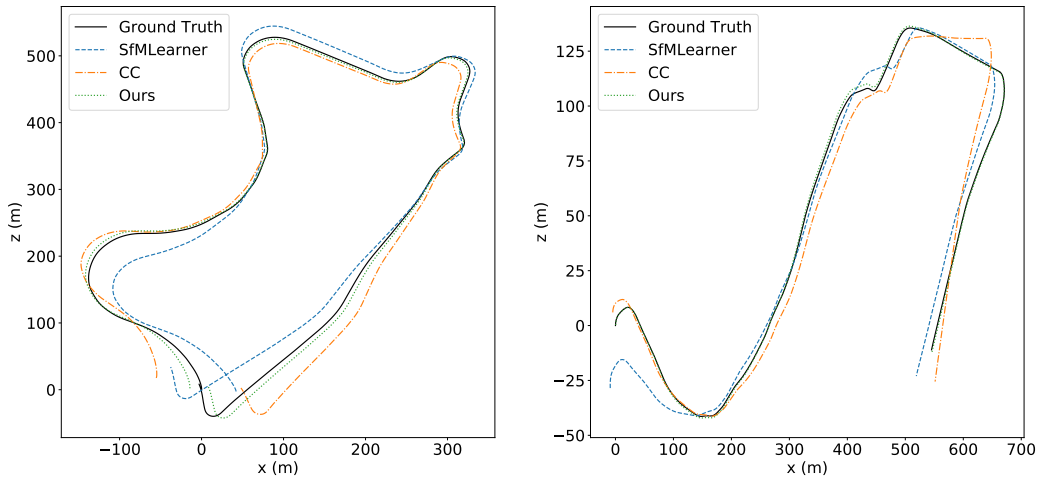


Figure 10.6: **Visual Trajectory on the KITTI VO dataset.** We compare our method against other deep learning based SfM methods on Seq.09 (Left) and Seq.10 (Right) of KITTI VO dataset.

Table 10.6: **Pose Estimation Results on MVS, Scenes11, and SUN3D Datasets**. The results of baselines SIFT and Matlab come from [98].

Method	MVS Dataset		Scenes11 Dataset		Sun3D Dataset	
	Pose		Pose		Pose	
	Rot	Tran	Rot	Tran	Rot	Tran
Base-SIFT	21.180	60.516	6.179	56.650	7.702	41.825
Base-Matlab	10.843	32.736	0.917	14.639	5.920	32.298
COLMAP [94]	7.961	23.469	4.834	10.682	4.235	15.956
DeMoN [98]	5.156	14.447	0.809	8.918	1.801	18.811
LS-Net [104]	4.653	11.221	4.653	8.210	1.521	14.347
BANet [103]	3.499	11.238	3.499	10.370	1.729	13.260
DeepSFM [107]	2.824	9.881	0.403	5.828	1.704	13.107
Ours	**2.417**	**3.878**	**0.276**	**2.041**	**1.391**	**10.757**

Table 10.7: **The Effect of Optical Flow Fine-tuning.** With the help of rigid flow supervision, our fine-tuned model achieves much better camera pose result than that of the pretrained model.

Model	MVS		Scenes11		Sun3D	
	Rot	Tran	Rot	Tran	Rot	Tran
Our-pretrain	3.637	10.984	0.587	6.617	1.670	12.905
Our-finetune	**2.417**	**3.878**	**0.276**	**2.041**	**1.391**	**10.757**

10.4.5 Framework Analysis and Justification

Estimating Camera Pose from Optical Flow. There are multiple ways to extract camera pose from optical flow. We consider two kinds of methods: deep regression and the classic five-point algorithm [245] with RANSAC scheme. For deep regression methods, we build a PoseNet similar to the one in [105] with ResNet50 [185] as the feature backbone, using image pairs and optical flow as the input. For the five-point algorithm, we use flow matching pairs as the input. We also set a baseline by using SIFT matches. To filter out error matches and outliers, we compare different masking strategies, such as flow uncertainty maps (output of per-pixel softmax operation), learned confidence maps, and SIFT feature locations.

We evaluate these methods on the MVS dataset, see Table 10.8. Deep regression methods have almost constant performance regardless of different inputs and masking strategies. Generally, methods using the classic five-point algorithm using dense optical flow with sparse masking outperform methods that perform deep regression. The best option is to use flow matches with masks based on SIFT feature locations.[4]

Dealing With Misaligned Scales. It is impossible to recover absolute scales from two-view images alone. This scale-ambiguity problem will cause trouble if we would like to directly use ground truth depth for supervision or through the widely used scale-invariant loss [65, 98]. We verify the effect of the proposed scale-invariant depth estimation module on the KITTI depth Eigen split. The baseline follows our pipeline but without scale-invariant depth module. It simply uses ℓ_{smooth} loss on the estimated

[4]Do not confuse SIFT feature detection, which we use to obtain state-of-the-art results, with SIFT feature matching, which performs poorly.

Table 10.8: **Estimating Camera Pose from Optical Flow.** We compare different methods to estimate camera pose from optical flow on the MVS dataset. "CNN" represents the pose regression network based on convolutional neural networks with ground truth pose supervision. "5-point" represents the five-point algorithm with RANSAC scheme. We also compare different flow masking strategies including flow uncertainty, flow confidence, and SIFT feature locations.

Method	Input	Sparse Mask	Rot	Tran
CNN	Color	-	6.652	17.834
CNN	Color + Flow	-	6.437	17.216
CNN	Color + Flow	Uncertainty	6.528	17.107
CNN	Color + Flow	Confidence	6.532	17.511
CNN	Color + Flow	SIFT loc	6.512	17.231
5-point	SIFT matches	-	10.622	29.731
5-point	Flow matches	-	15.673	37.292
5-point	Flow matches	Uncertainty	4.923	12.127
5-point	Flow matches	Confidence	4.614	11.022
5-point	Flow matches	SIFT Loc	**2.417**	**3.878**

depth and the ground depth regardless their scales, which forces the network to implicitly learn a scale.

As shown in Table 10.9, our scale-invariant depth module achieves a very similar accuracy to "Oracle Pose", which is the upper bound of our method. Its improvement over the baseline proves the effectiveness of our method. On the other hand, the performance of the scale-invariant loss is similar to the baseline method, which indicates that this loss cannot handle the scale problem.

Table 10.9: **Dealing with Misaligned Scales.** We compare different strategies to handle the misaligned scales between the estimated depth and ground truth depth on the KITTI Eigen split. 'Scale Inv Matching' indicates the scale invariant matching for plane sweeping, 'Scale Inv Loss' represents the scale invariant depth loss. The 'Oracle' means using the ground truth for both training and inference. Using ground truth pose for training achieves a worse result than the baseline, which verifies the scaling problem.

Strategy	Abs Rel	Sq Rel	RMSE	RMSE_{log}
Baseline	0.089	0.318	3.120	0.129
GT Pose Training	0.121	0.438	3.421	0.175
Scale Inv Loss	0.084	0.302	2.981	0.116
Scale Inv Matching	0.058	0.244	2.311	0.094
Oracle Scale	0.057	0.240	2.291	0.093
Oracle Pose	0.055	0.223	2.272	0.090

Random Dot SfM. In 1971, Bela Julesz proposes Random Dot Stereograms [163] to analyze human depth perceptions. Without giving any monocular cues, such as textures, semantics, shapes and *etc.*, the only way to estimate depth will be matching patterns from the random dot stereograms. Similarly, if a SfM network can recover depth maps from a random dot image pairs, it will be a key evidence to prove that such a network is based on matching to compute camera poses and depth maps.

To this end, we create a Random Dot SfM Dataset that contains 13,090 image pairs, 10,300 for training and 2,790 for testing. To generate the random dot image pairs, we first interpolate the sparse LiDAR points in KITTI VO dataset using [3] to get the dense depth maps and generate 13,090 random dot images as our reference frames. Then we use the ground truth camera poses from the KITTI VO dataset and the corresponding dense depth maps to generate the target frames for each random dot images. In this case, one can only recover the camera poses and dense depth maps through matching patterns between the random dot image pairs as there is no semantics, object shapes and textures provided to help the SfM process.

We exam our framework on random dot SfM images where there is no semantic content or image priors. After fine-tuning, our network can generate meaningful dense depth maps as shown in Fig. 10.7, which demonstrate that our framework is based on matching points to recover camera poses and dense depth maps.

Ref Image GT Ours

Figure 10.7: **Qualitative Result on the Random Dot SfM Dataset.** Our method is able to recover structure information from random dot image pairs, which indicates that our method does not rely on semantic contents or image priors.

Additional Open World Qualitative Result. We implement our framework in an autonomous driving car and test it in open world scenarios. As shown in Fig. 10.8, our framework is able to recover visually convincing depth maps, which demonstrates the generalization ability of our framework.

Reference Image Estimated Depth

Figure 10.8: **Depth Estimation on Open World Scenarios.**

10.5 Conclusions

In this chapter, we revisited the problem of deep neural network based two-view SfM and called for a return to the basics. First, we showed that existing deep learning-based SfM approaches tend to solve depth estimation and pose estimation as ill-posed problems, which overlooks the well-posedness nature of the SfM problem. Then we proposed a new deep two-view SfM framework that follows the classic SfM pipeline. Our framework consists of 1) an optical flow estimation module, 2) a relative pose estimation module, and 3) a scale-invariant depth estimation module. Extensive experiments showed that our proposed method outperforms all state-of-the-art two-view SfM methods on KITTI depth, KITTI VO, MVS, Scenes11, and SUN3D datasets in both pose estimation and depth estimation with a clear margin. In the future, we plan to extend our framework to other SfM problems such as three-view SfM and multi-view SfM, where the loop consistency and temporal consistency could further constrain these already well-posed problems.

Part VII

Conclusion

Chapter 11

Summary and Future Work

Visual 3D geometry recovery is a fundamental problem in computer vision. With several decades of research, this topic remains an active topic. The booming development of deep learning brings new blood into the field but many classic problems are yet to be solved, and new challenges remain.

11.1 Summary and Contributions

This thesis has been devoted to investigating self-supervised learning in visual 3D geometry recovery tasks. It addresses current challenges and pushes the limits of the state-of-the-art in various tasks, including depth completion, stereo matching, single mixture image depth estimation and optical flow. We summarize our contributions as follows.

For **depth completion** task, we proposed a global geometry constraint that enforces the reconstructed dense depth map as a weight sum of a set of principal components. A color-guided auto-regression model was added as a regularization term to make the generated depth map have sharp object boundaries. This proposed method can be efficiently solved in a closed form solution and outperforms previous traditional and deep learning based methods in various datasets. To deal with the artifacts in recovered depth map that caused by the texture of the color image, we leveraged the piecewise planar model for the estimated depth map and formulated it as a continuous Conditional Random Field optimization problem. It can be efficiently solved by TRW-s optimization.

For **stereo matching** task, we proposed a new self-supervised framework that can automatically adjust itself to open-world scenarios without using any ground truth labels. It targets the drawback of supervised deep methods that require a large number of ground truth labels for training and often have limited generalization capability. By assuming color consistency between stereo pairs, we utilized the image wrapping loss as our data term in the loss functions. Similar to traditional methods, we also use color-guided smoothness as our regularization term. In order to leverage the temporal consistency in stereo videos, convolutional-LSTM layers are applied in feature extraction and cost aggregation module. We extended the similar idea to LiDAR-Stereo fusion task, where we assume parts of the LiDAR points are

corrupted. We proposed a novel "feedback loop" to filter out the noise LiDAR points. It is achieved by checking the consistency between LiDAR points and LiDAR-Stereo fusion results. Moreover, we also proposed an efficient cost aggregation structure for stereo matching network that can process a pair of 540p images at 100 FPS.

We defined an original **single image depth estimation** task, where the input image is a mixture image of a stereo pair in a form of $I = \alpha I^{left} + (1 - \alpha) I^{right}$. Red-cyan, double vision and monocular image are three mixture types and we proposed a unified structure to deal all of them. Instead of brutal force regressing depth from the input image, we divide the task into two sub-tasks: view recovery and depth recovery. We first decouple the mixed image through image separation module and then do stereo matching on the separated pairs to get the depth map. The whole system only need original stereo pairs as supervisions and has better performance than previous methods.

For **optical flow** task, we proposed three ways to enforce global epipolar constraint and two of them can be applied to dynamic scenes. For stationary scenes, we first estimate a fundamental matrix from matching points and regularize the optical flow with the Sampson distance. For dynamic scenarios, we propose a low-rank constraint and a union-of-subspaces constraint. They avoid explicitly computing the fundamental matrix as well as multi-motion estimation.

For **structure from motion** task, we revisited the use of deep learning in two-view SfM, proposed a new deep two-view SfM framework that avoids ill-posedness. Our framework combines the best of deep learning and classical geometry. We also proposed a scale-invariant depth estimation module to handle the mismatched scales between ground truth depth and the estimated depth.

In the appendix, we propose a RGB-d semantic segmentation network as an application of leveraging geometry information for high level computer vision tasks.

11.2 Future Work

Multi-frame Structure From Motion.

In chapter 10, we proposed a deep two-view structure from motion framework that achieved state-of-the-art performance in term of camera pose estimation and dense depth estimation. However, in this framework, we did not consider multi-frame scenarios cases. How to extend the current framework to multi-frame scenarios is a non-trivial task. The key challenge for that is to have a consistent scale in all frames if we would like to bundle optimize multiple frames. Numerous of papers [258, 261] address the problem but none of them will work in our SfM framework due to the plane sweep process in depth estimation module. In the plane sweep process, it couples the matching process and depth estimation that one can only find the matching points with a given global scale while other methods can scale the depth independently (since they all use monocular depth estimation) to match the global scale without considering the matching points. The Trifocal Tensor [92] may be a direction to solve the chicken-egg problem in the deep multi-frame structure from

motion task.

Learning Generalization in Deep Neural Networks.

Visual geometry learning is often involved in several safety-critical applications like autonomous driving, so it is important to understand the actual behaviour of a deep network, and gain insights into the robustness. Using adversary attacks to analyze current deep networks is a way to test their robustness, but we could not know *why* they are robust or not. In Chapter 4, Chapter 5, and Chapter 10, we tested our networks with Random Dot image pairs to analyze the behaviour our networks. If our method can recover correct disparity/depth maps from random dot images, that means it does not rely on context information. Moreover, how to deal with domain gaps is also an important research direction in deep learning. We have proposed self-supervised learning methods to address this issue but there are still a large performance gap between supervised methods and self-supervised methods. This would be an interesting topic for future studies.

Part VIII

Appendix

Chapter 12

3D Geometry-Aware Semantic Labeling of Outdoor Street Scenes

This chapter is concerned with the problem of how to better exploit 3D geometric information for dense semantic image labeling. Existing methods often treat the available 3D geometry information (e.g., 3D depth-map) simply as an additional image channel besides the R-G-B colour channels, and apply the same technique for RGB image labeling. In this chapter, we demonstrate that directly performing 3D convolution in the framework of a residual connected 3D voxel top-down modulation network can lead to superior results. Specifically, we propose a 3D semantic labeling method to label outdoor street scenes whenever a dense depth map is available. Experiments on the "Synthia" and "Cityscape" datasets show our method outperforms the state-of-the-art methods, suggesting such a simple 3D representation is effective in incorporating 3D geometric information [1].

12.1 Introduction

Semantic labeling (semantic segmentation) aims to assign class labels (e.g., "cars", "road", "building", "pedestrian") to pixels in an image. It is an important task in computer vision and pattern recognition, which has found wide-range applications in the areas such as autonomous driving [40], robot SLAM[263], and augmented reality [264].

Deep Convolutional Neural Networks (CNNs) have gained tremendous success in almost all high-level vision tasks such as image classification, object detection, as well as semantic labeling [265][266][267]. The 2D convolution is defined in the image coordinate, where the filter is applied in the neighborhood defined by image pixel distance. Deep encoder-decoder (SegNet [266], dilated convolution (DeepLab-LargeFOV [267]) have also been proposed under the same framework. The success of these models mainly lies in their general modeling ability for complex unseen visual scenes.

To further improve the performance, deeper and wider networks [268] have been

[1] This work was originally published in [262]

proposed, which require massive labeled data during training. Even though these models have achieved state-of-the-art performance on various benchmarking datasets, they do not harness the full potentials of available depth/3D clues for semantic segmentation. Geometric information provides crucial and discriminative semantic cues for colour images. Depth maps generally provide complement information to colour images, where the 3D structure of the observed scene has been encoded naturally [269]. Therefore, semantic labeling will benefit from the availability of depth information. For indoor scenes, Hazirbas *et al.* [269] proposed a deep auto-encoder network for semantic labeling, where the encoder consists of two branches of networks that simultaneously extract features from colour and depth images and fuse depth features into the colour feature maps as the network goes deeper. Furthermore, Ma *et al.* [270] proposed to leverage the consistencies between multi-view semantic labeling.

However, most existing works have focused on indoor scene labeling where the size of the scene is limited. For outdoor street scene semantic labeling using depth information is difficult due to the following reasons: 1) difficulty in accurate depth acquisition for outdoor scenes; 2) large variation in scene scales; and 3) lacking of outdoor training datasets with dense depth information.

In this chapter, we advocate the benefit of using 3D information for outdoor labeling, and propose a simple and efficient way to use the 3D information. Specifically, we propose a direct way to represent RGB-D image in its natural 3D space, i.e., the way human sense the surrounding 3D world. Given a colour image and the associated depth map (from stereo vision or from LIDAR), we transform the colour image into 3D voxel space defined by the 3D position of each pixel, which enables subsequent 3D convolution to cater the 3D geometry in extracting semantic feature maps and thus achieves *3D geometry aware semantic labeling*.

To learn a geometry-aware representation, we propose a light-weight 3D Res-TDM (Residual connected Top-Down Modulation) structure that can squeeze 3D geometric information from depth map and own high resistance to noise and errors. We have performed experiments on the SYNTHIA dataset with ground truth depth map and the Cityscape dataset with computed disparity map. Experimental results demonstrate that our method outperforms the state-of-the-art semantic labeling methods, which indicates the success of our 3D voxel representation in effectively and efficiently encoding 3D geometric information.

The main contributions of our method can be summarized as:

1) A natural and direct 3D representation to encode RGB-D data, thus representing the semantic cues in 3D;
2) 3D convolution to exploit the geometric constraint for semantic labeling, enabling 3D geometry aware semantic labeling;
3) A light weighted 3D res-TDM structure that can squeeze 3D geometric information from depth map and own high resistance to noises and errors.

12.2 Related work

Semantic labeling: Before the era of deep learning, semantic segmentation has been widely formulated as CRF with hand-craft features and low-level vision cues. The breakthrough in deep learning has also been brought to semantic labeling to learn the nonlinear mapping from image to dense labeling in an end-to-end manner. The most noticeable deep convolutional network based semantic labeling method is FCN[265], which takes advantage of existing image classification architectures [8] [271] [185]. However, the decoder phase of FCN is relatively simple that makes it difficult to train. SegNet[266] tackles the above weakness by using an auto-encoder structure. Dilated convolution [267] has also been introduced to effectively enlarge the field of view of filters to incorporate larger context without increasing the number of parameters or the amount of computation.

RGB-D semantic labeling: Depth information has been used as an important cue to refine semantic labeling in computer vision [272]. Zhang *et al.* [273] designed hand-crafted depth features such as surface normals, height above ground and neighboring smoothness and put them into a classifier. Saurabh *et al.* [274] geocentrically encoded depth into disparity, height and angle as a HHA representation and proposed a 2.5D proposal for object detection and semantic segmentation. Lai *et al.* [275] utilizes HMP3D features in an MRF framework to label objects in 3D scenes. More recently, Li *et al.* [276] fused contextual information from RGB and depth channels by stacking convolutional layers with an LSTM layer, which memorizes both short- and long-range spatial dependencies in an image along vertical direction. Another LSTM-F layer has also been used to integrate contexts from different channels and bi-directional propagation is performed to fuse vertical contexts. Hazirbas *et al.* [269] proposed a simpler network with auto-encoder style, where two encoders are used to extract features from RGB image and depth image individually and one decoder is applied to decode RGB-D channels. Extracted depth features are fused with RGB in every encoder layer. In these works, colour features and depth features are coupled in a human-designed way, which may fail to exploit the strong correlation between colour image and depth map.

3D convolution: Volumetric (i.e., spatially 3D) convolution has been successfully used in video analysis ([277]). VoxNet [278] and 3D ShapeNet [279] are two pioneer works in applying 3D convolution on voxelized 3D shapes. Very recently, Song *et al.* [280] introduced 3D voxel representation of volumetric occupancy and simultaneously performed scene completion and scene parsing for indoor scenes. However, both works only preserve 3D structure information for object recognition and discard colour information in 3D convolution. Moreover, the output resolution of [280] is only $36 \times 60 \times 60$, which is insufficient for outdoor applications and it requires large labeled data for network training. Multi-view strategy has also been leveraged to exploit 3D geometry information. MVCNN [281] projects 3D point clouds onto different image planes and converts each view image into CNN features. However, this strategy could not be applied to outdoor semantic labeling task straightforwardly due to the difficulty in warping small objects between different views.

By contrast to the above works, we propose to make use of colour information as well as 3D structural information for dense semantic labeling under an unified framework. Our light-weight network architecture also allows us to increase the output dimension with a reasonable scale and can be trained from scratch with only thousands of samples.

12.3 Our Approach

Here, we describe our geometry-aware semantic labeling framework by performing 3D convolution in the framework of 3D voxel convolutional neural network. First, by contrast to existing methods that simply treating depth map as an additional channel besides the R-G-B channels, we represent the input RGB-D images in 3D voxel representation, where each voxel is associated with colour. Then a top-down module is proposed to exploit the rich 3D geometric information for outdoor scene semantic labeling, where 3D convolution is performed to extract 3D geometry aware features.

12.3.1 3D Representation

Given RGB-D images, existing methods either represent the generic 3D point clouds with volumetric or multi-view representation. The volumetric representation encodes a 3D shape as a 3D tensor of binary or real values while the multi-view representation encodes a 3D shape as a collection of renderings from multiple viewpoints. However these representations are mainly designed for indoor applications and cannot cope with outdoor scenarios for the following reasons : 1) difficulty in accurate depth acquisition; 2) large variation in scene scales; and 3) lack of outdoor training dataset with dense depth information. Furthermore, for a typical driving scene, the depth ranges from 0.5 meters to infinity (i.e., the sky), which makes it impossible to discretize depth values into a certain range. Therefore direct voxelizing in 3D space for outdoor street scene is infeasible.

To cater the above difficulties, we propose a new and yet direct 3D voxel representation for outdoor street scenes. Specifically, instead of resorting to the XYZ space for 3D point clouds, we propose to combine the image coordinate and the disparity directly, thus UVD space, where (U, V) index the 2D image coordinate while D indexes the discrete disparity. At a first glance, this 3D voxel representation may introduce severe distortion in 3D representation. Here we demonstrate that while providing simplicity in representation, the UVD 3D voxel representation also owns much desired geometric property as in the original XYZ space.

Theorem 1. *Any order curve in the XYZ 3D space corresponds to a 3D curve with the same order in the UVD 3D space.*

Proof. Without loss of generality, we take the second order surface in 3D as an example. A second order surface in the XYZ 3D space is defined by the following

equation:

$$[X_i, Y_i, Z_i, 1]\mathbf{A}[X_i, Y_i, Z_i, 1]^T = 0, \tag{12.1}$$

where (X_i, Y_i, Z_i) defines the 3D points on the surface and $\mathbf{A} \in \mathbb{R}^{4\times4}$ indexes the 3D surface. The UVD space and the XYZ space are connected via perspective projection:

$$X_i = \frac{(u_i - u_0)}{f_x} Z_i, Y_i = \frac{(v_i - v_0)}{f_y} Z_i, Z_i = \frac{fb}{d_i}, \tag{12.2}$$

where f_x, f_y, u_0, v_0 are the intrinsic parameters of the camera while f, b define the transformation from disparity d_i to 3D coordinate Z_i. By substituting these relations into the 3D surface and re-organizing the equation, we have

$$\begin{aligned} &\begin{bmatrix} u_i - u_0 \\ v_i - v_0 \\ 1, \\ d_i \end{bmatrix}^T \begin{bmatrix} \frac{fb}{f_x} & 0 & 0 & 0 \\ 0 & \frac{fb}{f_y} & 0 & 0 \\ 0 & 0 & fb & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \mathbf{A} \\ &\begin{bmatrix} \frac{fb}{f_x} & 0 & 0 & 0 \\ 0 & \frac{fb}{f_y} & 0 & 0 \\ 0 & 0 & fb & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_i - u_0 \\ v_i - v_0 \\ 1 \\ d_i \end{bmatrix} = 0. \end{aligned} \tag{12.3}$$

It is thus clear that a second order surface in XYZ space has been transformed to another second order surface in UVD space. The above proof could be extended to any order 3D surface directly. □

Therefore, we can conclude that any order parametric surfaces defined in the XYZ space have a corresponding surface of the same order in the UVD space. In other words, the transformation from XYZ space to UVD space is curve order preserving.

12.3.2 2D convolution VS 3D convolution

State-of-the-art semantic labeling methods use deep convolutional network to learn the nonlinear mapping from image to dense semantic labeling, where the convolution is conducted in a 2D manner. As the neighboring relation is defined on the image plane, the 2D convolution may fail to extract feature with 3D geometry aware. Instead, 3D convolution in the 3D voxel space could integrate the appearance cues in 3D geometry aware manner, *i.e.,* the 3D distance has been catered in convolution.

Given a colour image, the 2D convolution is expressed as Eq 12.4. The value of an unit at position (u, v) in the i^{th} feature map is denoted as $p_i^{u,v}$,

$$p_i^{uv} = \sum_k \sum_{m=0}^{M_i-1} \sum_{n=0}^{N_i-1} w_{ik}^{mn} p_{(i-1)k}^{(u+m)(v+n)}, \tag{12.4}$$

where w_{ik}^{mn} is the coefficient at the position (m, n) of the kernel connected to the k^{th} feature map. M, N are the height and width of the kernel. When the convolution is conducted in 3D, the value of an unit at position (u, v, d) in the i^{th} feature map denoted as $p_i^{u,v,d}$ is given by

$$p_i^{uvd} = \sum_k \sum_{m=0}^{M_i-1} \sum_{n=0}^{N_i-1} \sum_{l=0}^{L_i-1} w_{ik}^{mnl} p_{(i-1)k}^{(u+m)(v+n)(d+l)}, \tag{12.5}$$

where d is the third dimension of the feature map and L_i is the size of the 3D kernel along the third dimension. 3D convolution can extract features from both spatial and disparity dimensions.

In Fig. 12.1, we compare 2D convolution and 3D convolution for an outdoor street scene. As observed from the illustration, the 2D convolution extracts feature in neighborhood defined on the image plane, which could involve points far away in 3D space. By contrast, 3D convolution in the UVD space succeeds to extract features in a 3D geometry aware manner.

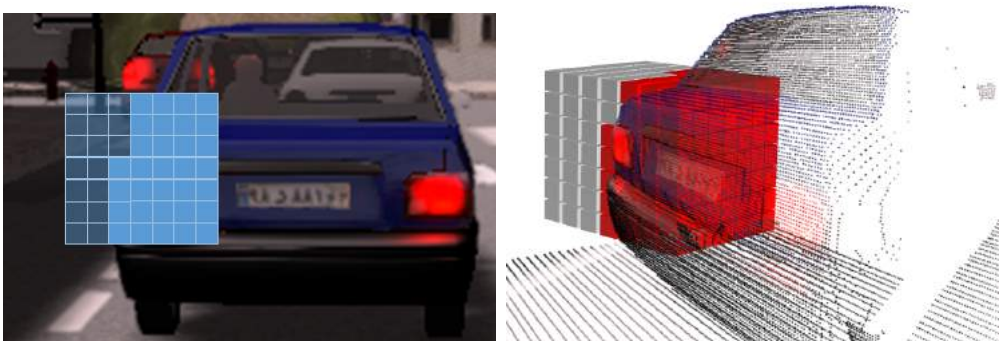


Figure 12.1: Illustration of 2D convolution and 3D convolution for semantic labeling. The left image demonstrates the widely used 2D convolution in extracting feature maps while the right image illustrates the corresponding 3D convolution conducted in the 3D voxel space. Note that the natural neighborhood relation is not preserved in the projection from 3D to 2D.

12.3.3 Network Architecture

Our goal is to assign each 3D point with a class label. A natural solution is to do 3D convolution on these point clouds. Also, since small objects such as traffic lights and signs play equally important role in semantic labeling, we adopt the idea of Top-Down Modulation (TDM) [282]. We not only convert it to 3D, but also modify it to better suit for our case. Note that in our 3D representation, most voxelized 3D labels are 0s. For these points, identity mappings are optimal. Therefore we swap the lateral connection between Bottom-up features and Top-down features with residual connection, and let the solvers simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings. Formally, we define a block from

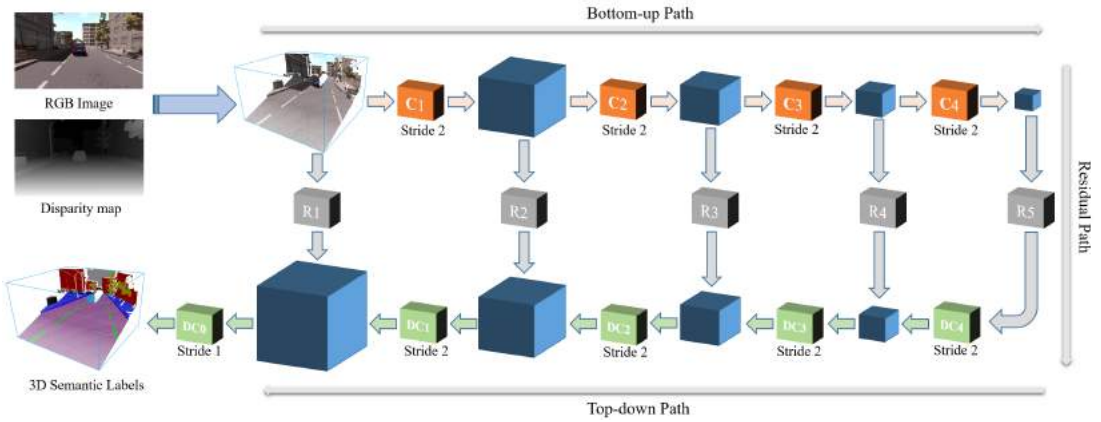


Figure 12.2: A nutshell of our 3D geometry-aware semantic labeling framework. The input RGB-D images are converted to the 3D voxel representation. C_i denotes the 3D convolution layer that encodes geometric and contextual information, R_i is residual module that connects low level features to the top-down pathway. DC_i is the 3D deconvolution layer to decode geometric and contextual information. We achieve a 3D semantic labeling, which could be projected to 2D for the sake of comparison.

Top-down path:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}_{C_i}, \{\mathbf{W}_i\}) + \mathbf{x}_{DC_i}, \tag{12.6}$$

where $\mathbf{y}$ denotes the output of residual connection, $\mathbf{x}_{C_i}$ is the output from the i-th convolution layer and $\mathbf{x}_{DC_i}$ is output from the i-th deconvolution layer. The function $\mathcal{F}(\mathbf{x}, \{\mathbf{W}_i\})$ is the residual mapping to learn. $+$ represents element-wise addition. Thus the dimensions of $\mathbf{x}_{R_i}$ and $\mathbf{x}_{DC_i}$ must be equal as in Eq. 12.6.

In Fig. 12.2, we present a nutshell of our overall architecture of the proposed network. Given a colour image and its corresponding disparity map, we first represent the RGB-D in the 3D UVD space as defined in Section 3.1. In the Bottom-up phase, the 3D volume ($H \times W \times (D+1) \times Ch$) passes through a series of 3D convolutional layers (C_i) with the same kernel size $3 \times 3 \times 3$ and a stride 2 until achieving an encoded feature volume with dimension $(1/16)H \times (1/16)W \times (1/16)(D+1) \times F$, where H, W, D, Ch, F represent the height, width, disparity levels, and number of channels and features respectively. In the Top-down phase, a mirrored process scales up the encoded feature volume back to the original size by swapping the 3D convolution with 3D deconvolution. For each scale, we apply our Res-TDM with a residual module R_i. Each R_i consists of two 3D convolution layers with the same kernel size $3 \times 3 \times 3$ and stride 1.

We employ the cross-entropy loss given by Eq. 12.7 as our loss function for training the network.

$$L(\mathbf{w}) = -\frac{1}{N+1}\sum_n [y_n \log \hat{y}_n + (1-y_n)\log(1-\hat{y}_n)] \tag{12.7}$$

where $\hat{y}_n = g(\mathbf{w}\dot{\mathbf{x}}_n)$ with logistic function $g(z)$. $\mathbf{w}$ is the vector of weights and each

sample is labeled by $n = 0, 1, 2, ..., N-1$. Note that there is a large variation in the number of pixels for each class. Despite of the imbalance distribution of valid labels, we only have $1/D$ valid labels in total. In other words, if the network predict all zeros, it can still achieve a training accuracy higher than 95%. In order to avoid our network drop to this local minimum, we apply a residual module R directly from the input that impose the network only to learn parameters around the areas with non-zero input.

Training our end-to-end pixel-wise semantic labeling network is very straightforward, which can be trained under the supervision of ground-truth semantic labels. Supervision is applied on the volumized 3D predictions and labels. All void 3D points (e.g., points before an object or behind an object) cannot be ignored and should be also given a void label 0. In the prediction phase, we perform max pooling along the disparity dimension to convert the 3D volume back to a 2D image and calculate the errors. This strategy can crease the robustness when dealing with noises and errors on disparity maps. For example, there is no guarantee that our network will predict the right label at the exact disparity level. When the disparity map is noisy, points with the same labels may have very different disparity levels. In this case, the network may predict the right label on the similar disparity level rather than the noise one.

12.4 Evaluation

In this section, we evaluate our proposed method with a comparison to alternative approaches and present an ablation study to better understand the proposed framework. Our method is evaluated on both synthetic and real datasets.

12.4.1 Dataset

For synthetic data, we use the SYNTHetic Collection of Imagery and Annotations (SYNTHIA) dataset [146], which contains 3 subsets: synthia-rand-cvpr16, synthia-rand-cityscapes and synthia-video-sequence. We choose the synthia-rand-cityscapes subset for experiments. It consists of 23 classes and a total of 9400 frames of outdoor scene with different weather and lighting conditions as well as randomly generated viewing angles. Since the dataset does not provide training and testing split, we randomly select 6000 frames for training, 1900 frames for validation and the remaining 1500 frames for testing. We manually convert the given ground truth depth maps to scaled inverse depth map in the range of $[0, 191]$ and resize the input image to 80×128.

For real data experiment, we use the Cityscape dataset [283], which contains 5,000 stereo frames of fine annotated ground truth semantic labels. We choose 2975 frames for training and use 500 frames for testing. In experiments, we compute the disparity map by using state-of-the-art stereo matching method, which is truncated to the range $[0, 111]$. The input images are resized to the resolution of 256×512.

Data augmentation We employ the mirror manipulation to augment the training examples for both datasets, since it maintains the geometry relationships.

12.4.2 Optimization

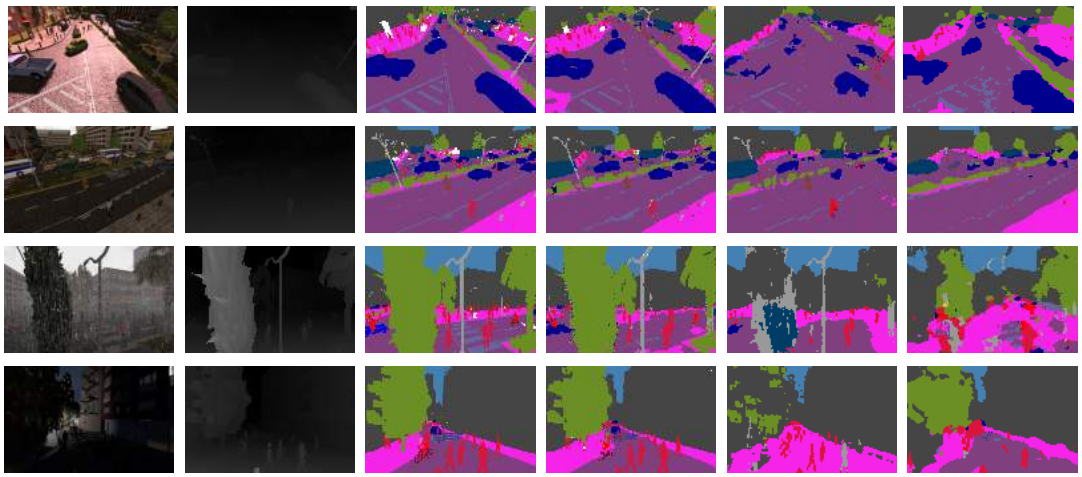
The proposed network architecture was implemented with Tensorflow [216]. We employed the RMSProp [217] with a constant learning rate of 1×10^{-3} to optimize all models in end-to-end manner. For the "Synthia" dataset, we normalized input images' RGB values to $[-1, 1]$ and trained our network from a random initialization for 50 epochs, which took 50 hours to converge by using a single NVIDIA Pascal Titan-X GPU and 1 second per frame in testing phase. However, the testing global accuracy climbs up to over 80% within one epoch. For the Cityscape dataset, we trained our network (S3D) with colour input for 30 epochs. In order to fit the 12G memory, we reduce the number of disparity levels to 48. We also trained our S3D network with feature input. The features were extracted from Resnet-38[268] with the same input dimension. The input feature dimension of our network is $32 \times 64 \times 512$. We added 3 extra upsampling layers in order to match the output resolution. The network converged quickly within 14 epochs. Note we did not use any post-processing to refine the results.

12.4.3 Evaluation metric

We measure the semantic labeling performance of our network with three metrics. Denote the total number of classes as k, p_{ij} as the amount of pixels belonging to class i which are predicted to be class j, the Global accuracy $G = \frac{\sum_i p_{ii}}{\sum_i \sum_j p_{ij}}$ measures the percentage of pixels correctly classified in the dataset. The Class Average Accuracy $C = \frac{1}{k+1} \sum_i \frac{p_{ii}}{\sum_j p_{ij}}$ normalizes the accuracy over the classes, therefore all classes share the same weight under this metric. Mean intersection over union $mIoU = \frac{1}{k+1} \sum_i \frac{p_{ii}}{\sum_j p_{ij} + \sum_j p_{ji} - p_{ij}}$ is used in the Cityscapes benchmark [283]. It is a more strict metric than class average accuracy since it penalizes false positive predictions.

12.4.4 Experimental results

Results on Synthia. In Table 12.1, we quantitatively compare our method (S3D) with state-of-the-art RGB semantic segmentation approach "SegNet" [266] and RGB-D based approach "FuseNet" [269]. For SegNet and FuseNet, we use the same input size and initialize the network parameters from the VGG model pre-trained on ImageNet. We train SegNet for 790 epochs and 230 epochs for FuseNet. For FuseNet, we use the same scaled inverse depth maps to train our network. Our method significantly outperforms competing methods with a notable margin under all three metrics: **18.6%** on class average accuracy, **7.6%** on global accuracy and **17.8%** on mIoU. Note that there are 4 classes never show up in testing set, so we remove them from the table and during the error calculation. In Fig. 12.3, we present qualitative



(a) colour image (b) Disparity (c) Ground truth (d) Our method (e) FuseNet (f) SegNet

Figure 12.3: **Quality comparison on the SYNTHIA dataset** We select images with different lighting and weather conditions as well as different viewing angles. Our method (d) shows superior performance, particularly it generates sharp boundaries for small objects. FuseNet (e) and SegNet (f) achieve similar performance but with the help of disparity map, FuseNet (e) captures more small objects such as pedestrians and poles.

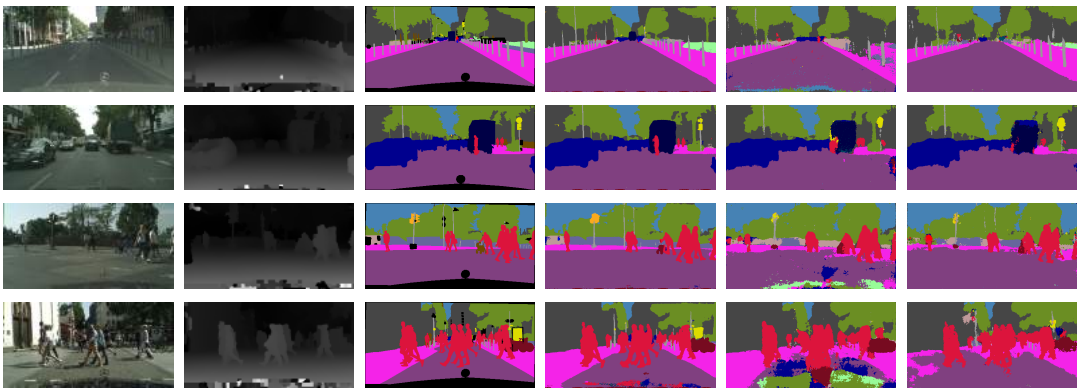
comparison between our method and state-of-the-art methods on the Synthia dataset, which clearly demonstrates the superior performance of our method.

Table 12.1: Performance evaluation on the SYNTHIA dataset

Method	sky	Building	Road	Sidewalk	Fence	Vegetation	Pole	Car	Traffic sign	Pedestrian	Bicycle	Motorcycle	Road-work	Traffic light	Rider	Bus	Wall	Lanemarking	Class avg.	Global avg.	mIoU
SegNet[266]	95.5	93.3	85.0	87.2	24.9	79.9	16.9	60.8	0.2	50.2	1.4	10.6	40.3	0.0	11.2	65.5	18.6	**45.9**	43.7	82.6	36.7
FuseNet[269]	92.4	94.5	79.9	70.2	35.6	73.0	29.9	64.4	2.5	57.5	2.8	9.4	46.4	2.6	16.4	60.9	13.7	20.9	42.9	78.1	35.9
S3D(ours)	**97.4**	**97.1**	**91.8**	**91.2**	**59.6**	**90.7**	**47.8**	**87.6**	**15.5**	**72.9**	**13.5**	**36.4**	**72.24**	**31.7**	**32.6**	**83.3**	**58.2**	42.1	**62.3**	**90.2**	**54.5**

Results on Cityscapes. Quantitative comparison with state-of-the-art semantic labeling methods on the Cityscapes dataset is shown in Table 12.2. The weight of FuseNet and SegNet are initialized from the VGG model trained on ImageNet. We also compare our method with the top performing one on the Cityscapes benchmark: ResNet-38[268]. Given RGB-D pair as input, we achieve similar performance with FuseNet. However, by swapping RGB image with trained features, our method outperforms all competing methods with a margin **1.2%**, **0.6%**, **1.0%** for class average accuracy, global accuracy and mIoU respectively. The margin is not as clear as previous one is due to the noise and errors in the disparity map. However, our algorithm still successfully squeezed useful information from it and increased the performance. Advanced disparity recovery algorithm [14] should lead to better performance. In Fig. 12.4, we present qualitative comparison between our framework and state-of-the-art methods on the Cityscapes dataset, which proves the superiority

of our method.



(a) colour image (b) Disparity (c) Ground truth (d) Our method (e) FuseNet (f) SegNet

Figure 12.4: **Qualitative evaluation on the Cityscapes dataset.** Our method with feature and disparity inputs (d) clearly outperforms the competing methods. The shape of pedestrians and poles are well preserved in our predictions. We shall see that FuseNet is effected by the noisy disparity map that has worse performance than SegNet. Note the invalid label is coloured with black.

Table 12.2: Performance evaluation on Cityscapes validation set

Method	Road	Sidewalk	Building	Wall	Fence	Pole	Traffic light	Traffic sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	Class avg.	Global avg.	mIoU
SegNet[266]	97.4	82.0	92.5	35.2	35.8	40.9	8.2	40.5	93.2	62.6	96.2	71.9	11.8	92.7	46.3	41.6	27.2	9.9	61.2	55.1	90.5	45.8
ResNet-38[268]	97.9	81.1	**93.6**	62.0	58.4	41.9	55.4	**62.5**	94.2	63.8	92.3	**79.1**	**54.9**	**94.7**	74.5	76.5	66.1	**61.4**	**75.1**	72.9	92.2	63.3
FuseNet[269]	88.6	82.1	93.3	25.4	48.2	42.7	0.5	47.0	94.5	38.5	**96.8**	75.8	0.5	93.6	56.9	2.2	12.4	0.0	60.2	50.5	87.4	39.1
S3D(RGB-d)	94.5	72.2	86.1	15.7	17.7	34.0	38.1	52.3	91.1	65.9	96.2	64.3	8.0	86.9	22.0	20.5	14.6	3.5	28.2	48.2	86.9	39.1
S3D(feature-d)	**98.0**	**87.4**	93.3	**66.2**	**71.3**	**46.8**	**60.2**	62.3	**95.2**	**67.4**	92.8	78.0	41.8	91.8	**78.7**	**81.2**	**73.3**	47.5	75.0	**74.1**	**92.8**	**64.3**

Ablation study To better understand the effectiveness of our 3D voxel representation, we perform ablation analysis and present the results in Table 12.3. S2D is the 2D version of our algorithm that replaces all 3D convolutions with 2D ones, where we stack the RGB image and disparity map into 4 channel input and plug into the S2D. S3D (Depth only) is the one with colourless "point clouds" which only provides shape information. According to this study, 3D voxel representation significantly improves the performance by 20.9% in mIoU.

Table 12.3: Ablation study on the SYNTHIA dataset

Method	sky	Building	Road	Sidewalk	Fence	Vegetation	Pole	Car	Traffic sign	Pedestrian	Bicycle	Motorcycle	Road-work	Traffic light	Rider	Bus	Wall	Lanemarking	Class avg.	Global avg.	mIoU
S2D(RGB-d)	98.0	92.0	62.6	82.2	41.6	80.9	33.4	68.6	2.5	66.9	0.3	8.9	60.2	0.9	1.6	37.0	0.0	16.6	41.9	77.9	33.6
S3D(d only)	**100.0**	**98.2**	91.4	**91.7**	59.5	**92.6**	**52.5**	83.8	0.3	**80.8**	12.6	2.8	46.4	0.7	25.9	76.0	38.0	0.1	52.9	89.9	47.0
S3D(RGB-d)	97.4	97.1	**91.8**	91.2	**59.6**	90.7	47.8	**87.6**	**15.5**	72.9	**13.5**	**36.4**	**72.24**	**31.7**	**32.6**	**83.3**	**58.2**	**42.1**	**62.3**	**90.2**	**54.5**

12.5 Conclusion

In this chapter, we have proposed a 3D voxel representation to integrate both appearance and depth information and a corresponding light-weight 3D Res-TDM network architecture for 3D geometry aware semantic segmentation. Our method provides an efficient and effective way to use geometric information to achieve better semantic labeling. Experiments on the "Synthia" and "Cityscape" datasets demonstrate that direct 3D convolution with our light-weight Res-TDM network can lead to superior performance, suggesting that such a simple 3D representation with Res-TDM is effective in incorporating 3D geometric information.

References

1. D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 1976. (cited on pages 3, 7, and 77)

2. K.S. Pennington and P.M. Will. A grid - coded technique for recording 3 - dimensional scenes illuminated with ambient light. *Optics Communications*, 2(4):167 – 169, 1970. (cited on page 3)

3. Xuelian Cheng, Yiran Zhong, Yuchao Dai, Pan Ji, and Hongdong Li. Noise-aware unsupervised deep lidar-stereo fusion. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 6339–6348, 2019. (cited on pages 3, 91, and 158)

4. Y. Y. Boykov and M. . Jolly. Interactive graph cuts for optimal boundary region segmentation of objects in n-d images. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 105–112 vol.1, 2001. (cited on page 4)

5. Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004. (cited on page 4)

6. H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003. (cited on page 4)

7. Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, February 2008. (cited on pages 4, 7, 48, 62, 63, 71, 78, and 94)

8. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Proc. Adv. Neural Inf. Process. Syst.*, pages 1097–1105. Curran Associates, Inc., 2012. (cited on pages 4 and 171)

9. Virginia R. DeSa. Learning classification with unlabeled data. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, page 112–119, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. (cited on pages 4 and 13)

10. Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016. (cited on pages 4, 14, and 113)

11. D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, 2016. (cited on pages 4 and 14)

12. Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Computer Vision – ECCV 2016*, pages 69–84, Cham, 2016. Springer International Publishing. (cited on pages 4 and 14)

13. Simon Jenni and Paolo Favaro. Self-supervised feature learning by learning to spot artifacts. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. (cited on pages 4 and 14)

14. Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv preprint arXiv:1709.00930*, 2017. (cited on pages 4, 7, 14, 49, 64, 66, 78, 93, 98, 100, 101, 114, 141, and 178)

15. Yiran Zhong, Hongdong Li, and Yuchao Dai. Open-world stereo video matching with deep rnn. In *Proc. Eur. Conf. Comp. Vis.*, September 2018. (cited on pages 4, 7, 47, 64, 78, 79, 93, and 96)

16. Yiran Zhong, Pan Ji, Jianyuan Wang, Yuchao Dai, and Hongdong Li. Unsupervised deep epipolar flow for stationary or dynamic scenes. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. (cited on pages 4, 125, and 141)

17. Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. (cited on pages 5 and 6)

18. P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. (cited on page 5)

19. D. Miao, J. Fu, Y. Lu, S. Li, and C. W. Chen. Texture-assisted kinect depth inpainting. In *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 604–607, 2012. (cited on page 5)

20. S. Liu, Y. Wang, J. Wang, H. Wang, J. Zhang, and C. Pan. Kinect depth restoration via energy minimization with tv21 regularization. In *2013 IEEE International Conference on Image Processing*, pages 724–724, 2013. (cited on page 5)

21. Chongyu Chen, Jianfei Cai, Jianmin Zheng, Tat Jen Cham, and Guangming Shi. Kinect depth recovery using a color-guided, region-adaptive, and depth-selective framework. *ACM Trans. Intell. Syst. Technol.*, 6(2), March 2015. (cited on page 5)

22. A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13(9):1200–1212, 2004. (cited on page 5)

23. S. Lu, X. Ren, and F. Liu. Depth enhancement via low-rank matrix completion. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3390–3397, 2014. (cited on page 5)

24. James Diebel and Sebastian Thrun. An application of markov random fields to range sensing. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 291–298, 2005. (cited on pages 6 and 20)

25. Henrik Andreasson, Rudolph Triebel, and Achim Lilienthal. *Non-Iterative Vision-Based Interpolation of 3D Laser Scans*, pages 83–90. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. (cited on pages 6 and 20)

26. Jennifer Dolson, Jongmin Baek, Christian Plagemann, and Sebastian Thrun. Upsampling range data in dynamic environments. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1141–1148. IEEE Computer Society, 2010. (cited on pages 6 and 20)

27. Jaesik Park, Hyeongwoo Kim, Yu-Wing Tai, Michael S. Brown, and Inso Kweon. High quality depth map upsampling for 3d-tof cameras. In *Proc. IEEE Int. Conf. Comp. Vis.*, ICCV '11, pages 1623–1630, Washington, DC, USA, 2011. IEEE Computer Society. (cited on pages 6 and 20)

28. J. Yang, X. Ye, K. Li, C. Hou, and Y. Wang. Color-guided depth recovery from rgb-d data using an adaptive autoregressive model. *IEEE Trans. Image Proc.*, 23(8):3443–3458, Aug 2014. (cited on pages 6, 20, 24, 25, 31, and 122)

29. S. Hawe, M. Kleinsteuber, and K. Diepold. Dense disparity maps from sparse disparity measurements. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2126–2133, Nov 2011. (cited on pages 6 and 20)

30. Y. Li, T. Xue, L. Sun, and J. Liu. Joint example-based depth map super-resolution. In *IEEE International Conference on Multimedia and Expo*, pages 152–157, July 2012. (cited on pages 6, 20, and 31)

31. X. Gong, J. Ren, B. Lai, C. Yan, and H. Qian. Guided depth upsampling via a cosparse analysis model. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 738–745, June 2014. (cited on pages 6 and 20)

32. Jonathan T. Barron and Ben Poole. *The Fast Bilateral Solver*, pages 617–632. Springer International Publishing, Cham, 2016. (cited on pages 6, 20, 31, 40, and 41)

33. Yu Li, Dongbo Min, Minh N. Do, and Jiangbo Lu. *Fast Guided Global Interpolation for Depth and Motion*, pages 717–733. Springer International Publishing, Cham, 2016. (cited on pages 6, 20, and 31)

34. Jason Ku, Ali Harakeh, and Steven L Waslander. In defense of classical image processing: Fast depth completion on the cpu. In *2018 15th Conference on Computer and Robot Vision (CRV)*, pages 16–22. IEEE, 2018. (cited on page 6)

35. Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision*, 2017. (cited on pages 6, 21, 92, 93, 97, 100, 101, and 104)

36. Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. *Learning a Deep Convolutional Network for Image Super-Resolution*, pages 184–199. Springer International Publishing, Cham, 2014. (cited on pages 6 and 21)

37. Song Xibin, Dai Yuchao, and Qin Xueying. Deep depth super-resolution : Learning depth super-resolution using deep convolutional neural network. In *Proc. Asian Conf. Comp. Vis.*, ACCV, 2016. (cited on pages 6, 21, and 28)

38. Gernot Riegler, Matthias Rüther, and Horst Bischof. *ATGV-Net: Accurate Depth Super-Resolution*, pages 268–284. Springer International Publishing, Cham, 2016. (cited on pages 6 and 21)

39. Maximilian Jaritz, Raoul De Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. Sparse and dense data with cnns: Depth completion and semantic segmentation. *arXiv preprint arXiv:1808.00769*, 2018. (cited on pages 6, 21, 93, and 97)

40. Nick Schneider, Lukas Schneider, Peter Pinggera, Uwe Franke, Marc Pollefeys, and Christoph Stiller. Semantically guided depth upsampling. In *German Conference on Pattern Recognition*, 2016. (cited on pages 6, 21, and 169)

41. Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, and Hongsheng Li. Depth completion from sparse lidar data with depth-normal constraints. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. (cited on page 6)

42. Jiaxiong Qiu, Zhaopeng Cui, Yinda Zhang, Xingdi Zhang, Shuaicheng Liu, Bing Zeng, and Marc Pollefeys. Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. (cited on page 6)

43. Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comp. Vis.*, 47(1-3):7–42, April 2002. (cited on pages 7, 47, 49, 63, 115, and 116)

44. M. Weber, M. Humenberger, and W. Kubinger. A very fast census-based stereo matching implementation on a graphics processing unit. In *IEEE 12th International Conference on Computer Vision Workshops*, pages 786–793, Sep. 2009. (cited on pages 7 and 63)

45. Chi Zhang, Zhiwei Li, Yanhua Cheng, Rui Cai, Hongyang Chao, and Yong Rui. Meshstereo: A global stereo model with mesh alignment regularization for view interpolation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2057–2065, 2015. (cited on pages 7, 63, and 91)

46. Tatsunori Taniai, Yasuyuki Matsushita, Yoichi Sato, and Takeshi Naemura. Continuous 3D Label Stereo Matching using Local Expansion Moves. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(11):2725–2739, 2018. (cited on pages 7 and 63)

47. Jure Žbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318, January 2016. (cited on pages 7, 49, 55, 56, 63, 65, 67, 71, 77, 79, 88, 93, 100, and 101)

48. Akihito Seki and Marc Pollefeys. Sgm-nets: Semi-global matching with neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017. (cited on pages 7, 49, and 63)

49. Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. StereoNet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *Proc. Eur. Conf. Comp. Vis.*, pages 8–14, 2018. (cited on pages 7, 63, 64, 65, 66, 71, 74, and 78)

50. Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. GA-Net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 185–194, 2019. (cited on pages 7, 62, 63, 64, 65, 66, 69, 71, 74, 75, 77, 78, 79, 83, 85, and 86)

51. N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2016. (cited on pages 7, 49, 54, 55, 56, 59, 62, 63, 64, 69, 71, 74, 77, 78, 79, 84, 85, 86, 87, 93, and 101)

52. Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2018. (cited on pages 7, 62, 63, 64, 71, 73, 74, 77, 85, 86, and 87)

53. Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. (cited on pages 7, 63, 64, 71, 75, and 76)

54. Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. (cited on pages 7 and 63)

55. Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. (cited on page 7)

56. Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 66–75, 2017. (cited on pages 7, 48, 49, 51, 52, 64, 65, 66, 71, 74, 77, 78, 79, 85, 86, and 97)

57. Nikolai Smolyanskiy, Alexey Kamenev, and Stan Birchfield. On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn. Workshops*, 2018. (cited on pages 7, 64, and 71)

58. Michael Bleyer and Christian Breiteneder. *Stereo Matching—State-of-the-Art and Research Challenges*, pages 143–179. Springer London, London, 2013. (cited on page 7)

59. A. M. Bronstein, M. M. Bronstein, M. Zibulevsky, and Y. Y. Zeevi. Sparse ica for blind separation of transmitted and reflected images. *Int'l J. Imaging Science and Technology*, 15(1):84–91, 2005. (cited on pages 8 and 110)

60. Jiaolong Yang, Hongdong Li, Yuchao Dai, and Robby T. Tan. Robust optical flow estimation of double-layer images under transparency or reflection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2016. (cited on pages 8, 110, 111, and 113)

61. Z. Li, P. Tan, R. T. Tan, D. Zou, Steven Zhiying Zhou, and L. F. Cheong. Simultaneous video defogging and stereo reconstruction. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4988–4997, June 2015. (cited on pages 8 and 110)

62. Armand Joulin and Sing Bing Kang. Recovering stereo pairs from anaglyphs. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 289–296, 2013. (cited on pages 9, 110, 111, 116, 117, 118, and 119)

63. Williem, R. Raskar, and I. K. Park. Depth map estimation and colorization of anaglyph images using local color prior and reverse intensity distribution. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 3460–3468, Dec 2015. (cited on pages 9, 110, 111, 116, 117, 118, and 119)

64. Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European Conference on Computer Vision*, 2016. (cited on page 9)

65. David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Proc. Adv. Neural Inf. Process. Syst.*, NIPS'14, pages 2366–2374, Cambridge, MA, USA, 2014. MIT Press. (cited on pages 9, 120, 122, 147, 150, 151, and 156)

66. Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. (cited on pages 10, 125, and 146)

67. Oisin Mac Aodha, Ahmad Humayun, Marc Pollefeys, and Gabriel J. Brostow. Learning a confidence measure for optical flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence (To Appear)*, 2012. (cited on page 10)

68. Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *Proc. Eur. Conf. Comp. Vis.*, pages 611–625. Springer, 2012. (cited on pages 10, 11, 127, 133, and 146)

69. Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *Int. J. Comp. Vis.*, 92(1):1–31, Mar 2011. (cited on pages 10 and 133)

70. Gilles Aubert, Rachid Deriche, and Pierre Kornprobst. Computing optical flow via variational techniques. *SIAM Journal on Applied Mathematics*, 60(1):156–182, 1999. (cited on pages 10 and 125)

71. Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*, pages 214–223. Springer, 2007. (cited on pages 10 and 125)

72. Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. Eur. Conf. Comp. Vis.*, pages 25–36. Springer, 2004. (cited on pages 10 and 125)

73. V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 508–515 vol.2, 2001. (cited on pages 10 and 125)

74. Moritz Menze, Christian Heipke, and Andreas Geiger. Discrete optimization for optical flow. In *German Conference on Pattern Recognition*, pages 16–28. Springer, 2015. (cited on pages 10 and 125)

75. Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2432–2439. IEEE, 2010. (cited on pages 10 and 125)

76. Jiaolong Yang and Hongdong Li. Dense, accurate optical flow estimation with piecewise parametric model. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1019–1027, 2015. (cited on pages 10 and 125)

77. Alexey Dosovitskiy, Philipp Fischery, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2758–2766, 2015. (cited on pages 10, 11, 125, 126, 133, and 146)

78. E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, Jul 2017. (cited on pages 10, 11, 125, 127, 133, and 134)

79. Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, July 2017. (cited on pages 10, 125, 126, and 134)

80. Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8934–8943, 2018. (cited on pages 10, 11, 125, 126, 127, 133, 134, and 144)

81. Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, page 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. (cited on page 10)

82. Connelly Barnes, Eli Shechtman, Dan B Goldman, and Adam Finkelstein. The generalized PatchMatch correspondence algorithm. In *European Conference on Computer Vision*, September 2010. (cited on page 11)

83. J. Lu, H. Yang, D. Min, and M. N. Do. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1854–1861, 2013. (cited on page 11)

84. L. Bao, Q. Yang, and H. Jin. Fast edge-preserving patchmatch for large displacement optical flow. *IEEE Transactions on Image Processing*, 23(12):4996–5006, 2014. (cited on page 11)

85. Levi Valgaerts, Andrés Bruhn, and Joachim Weickert. A variational model for the joint recovery of the fundamental matrix and the optical flow. In *Proceedings of DAGM Symposium on Pattern Recognition*, pages 314–324, 2008. (cited on pages 11 and 127)

86. A. Wedel, D. Cremers, T. Pock, and H. Bischof. Structure- and motion-adaptive regularization for high accuracy optic flow. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1663–1668, Sept 2009. (cited on pages 11 and 127)

87. K. Yamaguchi, D. McAllester, and R. Urtasun. Robust monocular epipolar flow estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1862–1869, June 2013. (cited on pages 11 and 127)

88. Jonas Wulff, Laura Sevilla-Lara, and Michael J. Black. Optical flow in mostly rigid scenes. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, July 2017. (cited on pages 11, 127, and 134)

89. Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2012. (cited on pages 11, 47, 54, 84, 99, 100, 127, and 133)

90. Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2015. (cited on pages 11, 34, 61, 69, 84, 115, 116, 127, and 133)

91. H Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981. (cited on pages 11, 143, and 146)

92. Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. (cited on pages 12, 78, 126, 128, 142, 144, 145, 146, and 164)

93. Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. (cited on pages 12, 141, and 144)

94. Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016. (cited on pages 12, 144, 154, and 156)

95. Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. (cited on pages 12 and 144)

96. Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. (cited on pages 12, 143, 144, and 146)

97. Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017. (cited on pages 13, 14, 49, 121, 122, 142, 145, 151, 152, and 154)

98. Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5038–5047, 2017. (cited on pages 13, 142, 145, 147, 151, 153, 154, and 156)

99. Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, volume 2, 2018. (cited on pages 13, 127, 134, 142, 145, and 152)

100. Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2018. (cited on pages 13 and 145)

101. Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 7063–7072, 2019. (cited on pages 13, 145, and 152)

102. Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J. Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. (cited on pages 13, 127, 134, 142, 145, 152, and 154)

103. Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *International Conference on Learning Representations*, 2018. (cited on pages 13, 142, 145, 149, 152, 154, and 156)

104. Ronald Clark, Michael Bloesch, Jan Czarnowski, Stefan Leutenegger, and Andrew J Davison. Ls-net: Learning to solve nonlinear least squares for monocular stereo. *arXiv preprint arXiv:1809.02966*, 2018. (cited on pages 13, 145, 154, and 156)

105. Kaixuan Wang and Shaojie Shen. Flow-motion and depth network for monocular stereo and beyond. *IEEE Robotics and Automation Letters*, 5(2):3307–3314, 2020. (cited on pages 13, 145, and 156)

106. Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *International Conference on Learning Representations*, 2020. (cited on pages 13, 142, 145, 147, 149, 151, and 152)

107. Xingkui Wei, Yinda Zhang, Zhuwen Li, Yanwei Fu, and Xiangyang Xue. Deepsfm: Structure from motion via deep bundle adjustment. *Proc. Eur. Conf. Comp. Vis.*, 2020. (cited on pages 13, 145, 147, 150, 153, 154, and 156)

108. J urgen Schmidhuber. Making the world differentiable: On using self-supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments. *Technical Report (TR FKI-126-90)*, 1989. (cited on page 13)

109. Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017. (cited on pages 14, 49, 93, 98, 109, 112, 115, 120, 121, and 122)

110. Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *European Conference on Computer Vision (ECCV) Workshops*, pages 3–10, 2016. (cited on pages 14, 125, 127, and 134)

111. Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *Proc. AAAI. Conf. Artificial Intelligence.*, volume 3, page 7, 2017. (cited on pages 14, 127, and 134)

112. Ravi Garg, BG Vijay Kumar, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *Proc. Eur. Conf. Comp. Vis.*, pages 740–756, 2016. (cited on pages 14, 49, 109, 112, 121, and 122)

113. Junyuan Xie, Ross Girshick, and Ali Farhadi. *Deep3D: Fully Automatic 2D-to-3D Video Conversion with Deep Convolutional Neural Networks*, pages 842–857. Springer International Publishing, Cham, 2016. (cited on pages 14, 49, and 112)

114. C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey. Learning Depth from Monocular Videos using Direct Methods. *ArXiv e-prints*, November 2017. (cited on pages 14 and 49)

115. Y. Luo, J. Ren, M. Lin, J. Pang, W. Sun, H. Li, and L. Lin. Single View Stereo Matching. *ArXiv e-prints*, March 2018. (cited on pages 14 and 49)

116. Meister Simon, Hur Junhwa, and Roth Stefan. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Proc. AAAI. Conf. Artificial Intelligence.*, AAAI'18, 2018. (cited on pages 14, 125, 127, 132, and 134)

117. Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2018. (cited on pages 14, 126, 127, 132, and 134)

118. Joel Janai, Fatma Güney, Anurag Ranjan, Michael J. Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *European Conference on Computer Vision (ECCV)*, volume Lecture Notes in Computer Science, vol 11220, pages 713–731. Springer, Cham, September 2018. (cited on pages 14, 126, 127, 134, and 135)

119. Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, October 2006. (cited on pages 15 and 34)

120. Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2147–2156, June 2016. (cited on page 19)

121. Joydeep Biswas and Manuela M. Veloso. Depth camera based indoor mobile robot localization and navigation. In *ICRA*, pages 1697–1702. IEEE, 2012. (cited on page 19)

122. Yi Zhou, Laurent Kneip, and Hongdong Li. Semi-dense visual odometry for rgb-d cameras using approximate nearest neighbour fields. In *ICRA*. IEEE, 2017. (cited on page 19)

123. Yu-I Yang, Chih-Kai Yang, and Chih-Hsing Chu. A virtual try-on system in augmented reality using rgb-d cameras for footwear personalization. *Journal of Manufacturing Systems*, 33(4):690 – 698, 2014. (cited on page 19)

124. Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *IEEE International Conference on Robotics and Automation*, pages 1–8. IEEE, 2018. (cited on pages 21, 92, 93, 97, 100, 101, and 104)

125. Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016. (cited on pages 21 and 93)

126. J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3431–3440, June 2015. (cited on page 21)

127. S. Hauberg, A. Feragen, and M. J. Black. Grassmann averages for scalable robust pca. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3810–3817, 2014. (cited on page 22)

128. Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. Eur. Conf. Comp. Vis.*, pages 746–760, Berlin, Heidelberg, 2012. Springer-Verlag. (cited on page 22)

129. Damien Garcia. Robust smoothing of gridded data in one and higher dimensions with missing values. In *Computational Statistics Data Analysis*, 2010. (cited on pages 22, 31, 32, 36, and 40)

130. David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Ruether, and Horst Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Proc. IEEE Int. Conf. Comp. Vis.*, December 2013. (cited on pages 27, 28, and 31)

131. Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *Proc. Eur. Conf. Comp. Vis.*, 2014. (cited on pages 29, 31, 55, 56, 100, 101, and 104)

132. Susstrunk Sabine, P. Fua, A. Shaji, A. Lucchi, K. Smith, and R. Achanta. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(undefined):2274–2282, 2012. (cited on pages 31 and 38)

133. Jian Peng, Tamir Hazan, David McAllester, and Raquel Urtasun. Convex max-product algorithms for continuous mrfs with applications to protein folding. In *Proc. Int. Conf. Mach. Learn.*, pages 729–736, 2011. (cited on page 34)

134. Frederic Besse, Carsten Rother, Andrew Fitzgibbon, and Jan Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. *Int. J. Comp. Vis.*, 110(1):2–13, October 2014. (cited on page 35)

135. Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Proc. Adv. Neural Inf. Process. Syst.*, NIPS'15, pages 802–810, Cambridge, MA, USA, 2015. MIT Press. (cited on pages 49 and 53)

136. Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *Arxiv*, 2017. (cited on pages 49 and 63)

137. W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5695–5703, June 2016. (cited on page 49)

138. Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *International Conf. on Computer Vision - Workshop on Geometry Meets Deep Learning*, 2017. (cited on pages 49, 62, 71, 73, and 74)

139. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, June 2016. (cited on page 49)

140. Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.*, 5(2):157–166, March 1994. (cited on page 53)

141. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. (cited on page 53)

142. Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 4346–4354, Washington, DC, USA, 2015. IEEE Computer Society. (cited on page 53)

143. Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Proc.*, 13(4):600–612, April 2004. (cited on pages 54 and 115)

144. D. Scharstein and C. Pal. Learning conditional random fields for stereo. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1–8, June 2007. (cited on pages 54, 57, and 58)

145. Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* IEEE Computer Society, 2007. (cited on pages 54, 57, and 58)

146. German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. (cited on pages 54, 56, 104, and 176)

147. Jianyuan Wang, Yiran Zhong, Yuchao Dai, Kaihao Zhang, Pan Ji, and Hongdong Li. Displacement-invariant matching cost learning for accurate optical flow estimation. In *Proc. Adv. Neural Inf. Process. Syst.*, 2020. (cited on pages 61, 76, 89, 143, 144, 146, and 149)

148. Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. (cited on page 61)

149. Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition*, pages 31–42, 2014. (cited on pages 61 and 84)

150. D. Hernandez-Juarez, A. Chacón, A. Espinosa, D. Vázquez, J.C. Moure, and A.M. López. Embedded real-time stereo estimation via semi-global matching on the GPU. *Procedia Computer Science*, 80:143–153, 2016. (cited on page 62)

151. Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. (cited on pages 64, 68, 71, 79, and 87)

152. Duggal Shivam, Wang Shenlong, Ma1 Wei-Chiu, Hu Rui, and Urtasun Raquel. DeepPruner: Learning efficient stereo matching via differentiable PatchMatch. In *Proc. IEEE Int. Conf. Comp. Vis.*, Nov 2019. (cited on pages 64, 71, 74, and 86)

153. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Proc. Eur. Conf. Comp. Vis.*, pages 346–361, 2014. (cited on page 65)

154. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015. (cited on pages 65 and 114)

155. Rohan Chabra, Julian Straub, Christopher Sweeney, Richard Newcombe, and Henry Fuchs. StereoDRNet: Dilated residual StereoNet. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. (cited on pages 66, 71, and 74)

156. Stepan Tulyakov, Anton Ivanov, and François Fleuret. Practical Deep Stereo (PDS): Toward applications-friendly deep stereo matching. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 5871–5881, 2018. (cited on page 71)

157. Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5410–5418, 2018. (cited on pages 71, 74, 78, 79, 85, 86, 87, 97, 100, and 101)

158. Xinjing. Cheng, Peng. Wang, and Ruigang. Yang. Learning depth with convolutional spatial propagation network. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1–1, 2019. (cited on pages 71 and 74)

159. Guang-Yu Nie, Ming-Ming Cheng, Yun Liu, Zhengfa Liang, Deng-Ping Fan, Yue Liu, and Yongtian Wang. Multi-level context ultra-aggregation for stereo matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. (cited on page 71)

160. A. Kuzmin, D. Mikushin, and V. Lempitsky. End-to-end learning of cost-volume aggregation for real-time dense stereo. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Sep. 2017. (cited on page 71)

161. H. Lee and Y. Shin. Real-time stereo matching network with high accuracy. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4280–4284, Sep. 2019. (cited on page 71)

162. Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *Proc. Eur. Conf. Comp. Vis.*, pages 626–643, 2018. (cited on page 71)

163. Bela Julesz. *Foundations of Cyclopean Perception*. Chicago: The University of Chicago Press, 1971. (cited on pages 75 and 157)

164. Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. (cited on page 75)

165. Rolf D. Henkel. Constructing the cyclopean view. In *Artificial Neural Networks — ICANN'97*, pages 907–912, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg. (cited on pages 75 and 76)

166. Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Hongdong Li, Tom Drummond, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. *Advances in Neural Information Processing Systems*, 33, 2020. (cited on page 77)

167. Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *Proc. Int. Conf. Learning Representations*, 2019. (cited on page 78)

168. Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 10734–10742, 2019. (cited on page 78)

169. Jiemin Fang, Yuzhu Sun, Qian Zhang, Yuan Li, Wenyu Liu, and Xinggang Wang. Densely connected search space for more flexible neural architecture search. *arXiv preprint arXiv:1906.09607*, 2019. (cited on page 78)

170. Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. *arXiv preprint arXiv:1911.09070*, 2019. (cited on page 78)

171. Junran Peng, Ming Sun, ZHAO-XIANG ZHANG, Tieniu Tan, and Junjie Yan. Efficient neural architecture transformation search in channel-level for object detection. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 14290–14299, 2019. (cited on page 78)

172. Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 82–92, 2019. (cited on pages 78, 79, 80, 82, 83, 84, and 87)

173. Yiheng Zhang, Zhaofan Qiu, Jingen Liu, Ting Yao, Dong Liu, and Tao Mei. Customizable architecture search for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 11641–11650, 2019. (cited on pages 78 and 80)

174. Wuyang Chen, Xinyu Gong, Xianming Liu, Qian Zhang, Yuan Li, and Zhangyang Wang. Fasterseg: Searching for faster real-time semantic segmentation. *International Conference on Learning Representation*, 2020. (cited on pages 78, 79, and 80)

175. Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. (cited on page 78)

176. Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8697–8710, 2018. (cited on page 78)

177. Krizhevsky Alex. Learning multiple layers of features from tiny images. In *Tech Report*, 2009. (cited on page 79)

178. Tonmoy Saikia, Yassine Marrakchi, Arber Zela, Frank Hutter, and Thomas Brox. Autodispnet: Improving disparity estimation with automl. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1812–1823, 2019. (cited on pages 79, 80, 81, 82, 85, 86, 87, 88, and 89)

179. Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2020. (cited on pages 79, 85, 86, 87, and 89)

180. Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 4053–4061. Curran Associates, Inc., 2016. (cited on page 80)

181. Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. In *Proc. Brit. Mach. Vis. Conf.*, 2017. (cited on page 80)

182. Liang-Chieh Chen, Maxwell Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jon Shlens. Searching for efficient multi-scale architectures for dense image prediction. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 8699–8710, 2018. (cited on page 80)

183. Vladimir Nekrasov, Hao Chen, Chunhua Shen, and Ian Reid. Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 9126–9135, 2019. (cited on page 80)

184. Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. (cited on pages 81, 82, and 87)

185. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. (cited on pages 82, 156, and 171)

186. Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. In *International Conference on Learning Representations*, 2020. (cited on page 88)

187. Y. Dai, Z. Zhu, Z. Rao, and B. Li. MVS2: Deep unsupervised multi-view stereo with multi-view symmetry. In *International Conference on 3D Vision*, pages 1–8, 2019. (cited on page 89)

188. Jonathan Kelly and Gaurav S Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *International Journal of Robotics Research*, 30(1):56–79, 2011. (cited on page 91)

189. Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168. IEEE, 2011. (cited on page 91)

190. Florin Oniga and Sergiu Nedevschi. Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection. *IEEE Transactions on Vehicular Technology*, 59(3):1172–1182, 2010. (cited on page 91)

191. Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium*, pages 963–968. Ieee, 2011. (cited on page 91)

192. Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comp. Vis.*, 47(1-3):7–42, 2002. (cited on page 91)

193. Jeffrey S Deems, Thomas H Painter, and David C Finnegan. Lidar measurement of snow depth: a review. *Journal of Glaciology*, 59(215):467–479, 2013. (cited on page 91)

194. H. Badino, D. Huber, and T. Kanade. Integrating lidar into stereo for fast and improved disparity computation. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 405–412, May 2011. (cited on pages 91 and 93)

195. W. Maddern and P. Newman. Real-time probabilistic fusion of sparse 3d lidar and dense stereo. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2181–2188, Oct 2016. (cited on pages 91, 94, 100, and 101)

196. Kihong Park, Seungryong Kim, and Kwanghoon Sohn. High-precision depth estimation with the 3d lidar and stereo fusion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2156–2163. IEEE, 2018. (cited on pages 91, 94, 100, and 101)

197. Yinda Zhang, Sameh Khamis, Christoph Rhemann, Julien Valentin, Adarsh Kowdle, Vladimir Tankovich, Michael Schoenberg, Shahram Izadi, Thomas Funkhouser, and Sean Fanello. Activestereonet: End-to-end self-supervised learning for active stereo systems. In *Proc. Eur. Conf. Comp. Vis.*, September 2018. (cited on page 93)

198. Nathaniel Chodosh, Chaoyang Wang, and Simon Lucey. Deep convolutional compressed sensing for lidar depth completion. *arXiv preprint arXiv:1803.08949*, 2018. (cited on page 93)

199. Samir El-Omari and Osama Moselhi. Integrating 3d laser scanning and photogrammetry for progress measurement of construction work. *Automation in Construction*, 18(1):1 – 9, 2008. (cited on page 93)

200. Peyman Moghadam, Wijerupage Sardha Wijesoma, and Dong Jun Feng. Improving path planning and mapping based on stereo vision and lidar. In *International Conference on Control, Automation, Robotics and Vision*, pages 384–389. IEEE, 2008. (cited on page 93)

201. Kevin Nickels, Andres Castano, and Christopher M. Cianci. Fusion of lidar and stereo range for mobile robots. *International Conference on Advanced Robotics (ICAR)*, 1:65–70, 2003. (cited on page 93)

202. Alastair Harrison and Paul Newman. Image and sparse laser fusion for dense scene reconstruction. In Andrew Howard, Karl Iagnemma, and Alonzo Kelly, editors, *Field and Service Robotics*, pages 219–228, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. (cited on page 93)

203. V. Gandhi, J. Čech, and R. Horaud. High-resolution depth maps based on tof-stereo fusion. In *IEEE International Conference on Robotics and Automation*, pages 4742–4749, May 2012. (cited on page 93)

204. Nick Schneider, Lukas Schneider, Peter Pinggera, Uwe Franke, Marc Pollefeys, and Christoph Stiller. Semantically guided depth upsampling. In *German Conference on Pattern Recognition*, pages 37–48. Springer, 2016. (cited on pages 99 and 100)

205. Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. (cited on page 99)

206. Yiran Zhong, Yuchao Dai, and Hongdong Li. Stereo computation for a single mixture image. In *Proc. Eur. Conf. Comp. Vis.*, September 2018. (cited on page 109)

207. Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2015. (cited on pages 109 and 120)

208. A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):228–242, Feb 2008. (cited on page 110)

209. Yu Li and Michael S. Brown. Single image layer separation using relative smoothness. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2752–2759, 2014. (cited on pages 111 and 120)

210. Qingnan Fan, Jiaolong Yang, Gang Hua, Baoquan Chen, and David Wipf. A generic deep architecture for single image reflection removal and image smoothing. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2017. (cited on page 111)

211. Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, May 2011. (cited on page 111)

212. Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Trans. Graph.*, 35(4):110:1–110:11, July 2016. (cited on page 113)

213. Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017. (cited on page 113)

214. A. Levin and Y. Weiss. User assisted separation of reflections from a single image using a sparsity prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(9):1647–1654, September 2007. (cited on page 113)

215. Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Proc. Adv. Neural Inf. Process. Syst.*, pages 2017–2025. Curran Associates, Inc., 2015. (cited on page 115)

216. Martín Abadi, Ashish Agarwal, and Paul Barham *et al.* . Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016. (cited on pages 115 and 177)

217. Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012. (cited on pages 115 and 177)

218. Bo Li, Yuchao Dai, and Mingyi He. Monocular depth estimation with hierarchical fusion of dilated cnns and soft-weighted-sum inference. *Pattern Recognition*, 83:328 – 339, 2018. (cited on page 120)

219. Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 2017–2025, 2015. (cited on page 126)

220. Jean-Yves Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001. (cited on page 126)

221. Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992. (cited on page 126)

222. Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. An improved algorithm for tv-l 1 optical flow. In *Statistical and geometrical approaches to visual motion analysis*, pages 23–45. Springer, 2009. (cited on page 126)

223. Luis Alvarez, Rachid Deriche, Théo Papadopoulo, and Javier Sánchez. Symmetrical dense optical flow estimation with occlusions detection. *Int. J. Comp. Vis.*, 75(3):371–385, 2007. (cited on page 126)

224. Ravi Garg, Luis Pizarro, Daniel Rueckert, and Lourdes Agapito. Dense multiframe optic flow for non-rigid objects using subspace constraints. In *Proc. Asian Conf. Comp. Vis.*, pages 460–473, 2011. (cited on page 127)

225. Yuliang Zou, Zelun Luo, and Jia-Bin Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *Proc. Eur. Conf. Comp. Vis.*, pages 38–55, 2018. (cited on pages 127, 134, and 152)

226. Pan Ji, Hongdong Li, Mathieu Salzmann, and Yiran Zhong. Robust multi-body feature tracker: a segmentation-free approach. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3843–3851, 2016. (cited on page 129)

227. Benjamin Recht, Weiyu Xu, and Babak Hassibi. Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. In *IEEE Conference on Decision and Control*, pages 3065–3070. IEEE, 2008. (cited on page 129)

228. Zhuwen Li, Jiaming Guo, Loong-Fah Cheong, and Steven Zhiying Zhou. Perspective motion segmentation via collaborative clustering. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1369–1376, 2013. (cited on page 130)

229. Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013. (cited on page 130)

230. Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *Proc. Eur. Conf. Comp. Vis.*, pages 347–360. Springer, 2012. (cited on page 130)

231. Pan Ji, Mathieu Salzmann, and Hongdong Li. Efficient dense subspace clustering. In *IEEE Winter Conference on Applications of Computer Vision*, pages 461–468. IEEE, 2014. (cited on pages 130 and 131)

232. Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):171–184, 2013. (cited on page 130)

233. Ehsan Elhamifar and René Vidal. Clustering disjoint subspaces via sparse representation. In *IEEE International Conference on Acoustics Speech and Signal Processing*, pages 1926–1929. IEEE, 2010. (cited on page 131)

234. Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2015. (cited on page 134)

235. A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007. (cited on page 141)

236. Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007. (cited on page 141)

237. Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *ICRA*, 2014. (cited on page 141)

238. J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, 2014. (cited on page 141)

239. J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. In *arXiv:1607.02565*, 2016. (cited on page 141)

240. Shinya Sumikura, Mikiya Shibuya, and Ken Sakurada. OpenVSLAM: A versatile visual SLAM framework. In *ACM Multimedia Open Source Software Competition*, 2019. (cited on page 141)

241. M. Bertero, T. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76(8):869–889, August 1988. (cited on page 142)

242. T. Hassner and R. Basri. Example based 3D reconstruction from single 2D images. In *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2006. (cited on page 142)

243. Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2Vox: Context-aware 3D reconstruction from single and multi-view images. In *ICCV*, 2019. (cited on page 142)

244. Kejie Li, Ravi Garg, Ming Cai, and Ian Reid. Single-view object shape reconstruction using deep shape prior and silhouette. In *BMVC*, 2019. (cited on page 142)

245. Hongdong Li and Richard Hartley. Five-point motion estimation made easy. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 1, pages 630–633. IEEE, 2006. (cited on pages 143, 146, and 156)

246. Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. *arXiv preprint arXiv:2003.12039*, 2020. (cited on page 144)

247. Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. (cited on pages 146 and 150)

248. Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4040–4048, 2016. (cited on page 146)

249. David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999. (cited on page 146)

250. David Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004. (cited on pages 146 and 150)

251. Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. *International Conference on Learning Representations*, 2019. (cited on pages 147 and 149)

252. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008. (cited on page 149)

253. Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005. (cited on page 149)

254. Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *Proceedings of the IEEE international conference on computer vision*, pages 1625–1632, 2013. (cited on page 151)

255. Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2002–2011, 2018. (cited on pages 151 and 152)

256. Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 5684–5693, 2019. (cited on pages 151 and 152)

257. Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *Proc. IEEE Int. Conf. Comp. Vis.*, October 2019. (cited on pages 151 and 153)

258. Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Advances in neural information processing systems*, pages 35–45, 2019. (cited on pages 154 and 164)

259. Huangying Zhan, Chamara Saroj Weerasekera, Jiawang Bian, and Ian Reid. Visual odometry revisited: What should be learnt? *arXiv preprint arXiv:1909.09803*, 2019. (cited on page 154)

260. Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. In *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 376–380. IEEE, 1991. (cited on page 155)

261. Yuliang Zou, Pan Ji, , Quoc-Huy Tran, Jia-Bin Huang, and Manmohan Chandraker. Learning monocular visual odometry via self-supervised long-term modeling. In *Proc. Eur. Conf. Comp. Vis.*, 2020. (cited on page 164)

262. Yiran Zhong, Yuchao Dai, and Hongdong Li. 3d geometry-aware semantic labeling of outdoor street scenes. In *Proc. Int. Conf. Pattern Recogn.*, pages 2343–2349. IEEE, 2018. (cited on page 169)

263. Javier Civera, Dorian Gálvez-López, L. Riazuelo, Juan D. Tardós, and J. M. M. Montiel. Towards semantic slam using a monocular camera. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1277–1284, Sept 2011. (cited on page 169)

264. Tamás Matuszka, Gergő Gombos, and Attila Kiss. A new approach for indoor navigation using semantic webtechnologies and augmented reality. In *International Conference on Virtual, Augmented and Mixed Reality*, pages 202–210. Springer, 2013. (cited on page 169)

265. Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional models for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2015. (cited on pages 169 and 171)

266. Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. (cited on pages 169, 171, 177, 178, and 179)

267. Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *Proc. Int. Conf. Learning Representations*, 2015. (cited on pages 169 and 171)

268. Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. arXiv:1611.10080, 2016. (cited on pages 169, 177, 178, and 179)

269. Caner Hazirbas, Lingni Ma, Csaba Domokos, and Daniel Cremers. Fusenet: incorporating depth into semantic segmentation via fusion-based cnn architecture. In *Asian Conference on Computer Vision*, November 2016. (cited on pages 170, 171, 177, 178, and 179)

270. Lingni Ma, Jörg Stückler, Christian Kerl, and Daniel Cremers. Multi-view deep learning for consistent semantic mapping with RGB-D cameras. *CoRR*, abs/1703.08866, 2017. (cited on page 170)

271. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. (cited on page 171)

272. Andreas C. Müller and Sven Behnke. Learning depth-sensitive conditional random fields for semantic segmentation of rgb-d images. In *IEEE International Conference on Robotics and Automation*, pages 6232–6237, May 2014. (cited on page 171)

273. Chenxi Zhang, Liang Wang, and Ruigang Yang. Semantic segmentation of urban scenes using dense depth maps. In *Proc. Eur. Conf. Comp. Vis.*, pages 708–721, Berlin, Heidelberg, 2010. Springer-Verlag. (cited on page 171)

274. Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. *Learning Rich Features from RGB-D Images for Object Detection and Segmentation*, pages 345–360. ECCV, Cham, 2014. (cited on page 171)

275. K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057, 2014. (cited on page 171)

276. Zhen Li, Yukang Gan, Xiaodan Liang, Yizhou Yu, Hui Cheng, and Liang Lin. RGB-D scene labeling with long short-term memorized fusion model. *CoRR*, abs/1604.05000, 2016. (cited on page 171)

277. Abhijit Kundu, Vibhav Vineet, and Vladlen Koltun. Feature space optimization for semantic video segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3168–3175, June 2016. (cited on page 171)

278. Daniel Maturana and Sebastian Scherer. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In *IROS*, 2015. (cited on page 171)

279. Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2015. (cited on page 171)

280. Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017. (cited on page 171)

281. Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2015. (cited on page 171)

282. Abhinav Shrivastava, Rahul Sukthankar, Jitendra Malik, and Abhinav Gupta. Beyond skip connections: Top-down modulation for object detection. *CoRR*, abs/1612.06851, 2016. (cited on page 174)

283. Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016. (cited on pages 176 and 177)