# IMPROVING 3D RECONSTRUCTION USING DEEP LEARNING PRIORS

Rohan Chabra

A dissertation submitted to the faculty at the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2020

Approved by:

Henry Fuchs

Aniket Bera

Gary Bishop

Jan-Michael Frahm

Richard Newcombe

# ABSTRACT

Rohan Chabra: IMPROVING 3D RECONSTRUCTION USING DEEP LEARNING PRIORS
(Under the direction of Henry Fuchs)

Modeling the 3D geometry of shapes and the environment around us has many practical applications in mapping, navigation, virtual/ augmented reality, and autonomous robots. In general, the acquisition of 3D models relies on using passive images or using active depth sensors such as structured light systems that use external infrared projectors. Although active methods provide very robust and reliable depth information, they have limited use cases and heavy power requirements, which makes the passive techniques more suitable for day-to-day user applications. Image-based depth acquisition systems usually face challenges representing thin, textureless, or specular surfaces and regions in shadows or low-light environments. While scene depth information can be extracted from the set of passive images, fusion of depth information from several views into a consistent 3D representation remains a challenging task. The most common challenges in 3D environment capture include the use of efficient scene representation that preserves the details, thin structures, and ensures overall completeness of the reconstruction.

In this thesis, we illustrate the use of deep learning techniques to resolve some of the challenges of image-based depth acquisition and 3D scene representation. We use a deep learning framework to learn priors over scene geometry and scene global context for solving several ambiguous and ill-posed problems such as estimating depth on textureless surfaces and producing complete 3D reconstruction for partially observed scenes. More specifically, we propose that using deep learning priors, a simple stereo camera system can be used to reconstruct a typical apartment size indoor scene environments with the fidelity that approaches the quality of a much more expensive state-of-the-art active depth-sensing system. Furthermore, we describe how deep learning priors on local shapes can represent 3D environments more efficiently than with traditional systems while at the same time preserving details and completing surfaces.

*To my parents*

# ACKNOWLEDGEMENTS

There are key individuals that have provided me with the support and tools that a graduate student could want. Here, I wish to acknowledge those friends, advisors, and relatives that have guided and inspired me throughout this doctoral endeavour.

First and foremost, I would like to thank my supervisor, Prof. Henry Fuchs for giving me an opportunity to be part of a unique research experience under his guidance. His experience, expertise, and thoroughness is largely responsible for the quality of research in my dissertation. I would like to thank my loving fiancée, Nirupama Sharma for being very supportive and patient with me during my every struggle as a doctorate student. I am extremely grateful to my dearest friend Aniket Bera for helping me throughout my graduate student life. He has been the source of motivation for my pursuit to doctorate and has helped and guided me in achieving several milestones during my doctoral journey.

I am extremely grateful to Dr. Richard Newcombe for giving me an excellent opportunity to work with him and his team at Facebook Reality Labs during the course of several internships and research collaborations. This fortunate collaboration with Richard gave me a chance to use the state-of-the-art research tools and an opportunity to work with an expert research team. Richard, also guided me in the development of several innovative ideas and technical concepts that became an integral part of this dissertation. This dissertation would not have been possible without his immense support and guidance.

I am thankful to my father, Sanjay Chhabra, and my mother, Reena Chhabra for always being supportive of me during my entire student life. They have always strengthen me with motivation and encouragement during several difficult phases of my life. I would also like to thank my dearest friends whom I met at Chapel Hill, Tanmay Randhavane, Nicholas Rewkowski, Lakshita Jain, Akash Bapat, Srihari Pratapa, and many others for their support. I thank my colleagues and friends at Facebook, Julian Straub, Jan Eric Lenssen, Lingni Ma, Eddy Ilg, Tanner Schmidt, and Steven Lovegrove for helping me with my several internship projects.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| SDF | Signed Distance Function |
| TSDF | Truncated Signed Distance Function |
| NCC | Normalized Cross Correlation |
| ZNCC | Zero Normalized Cross Correlation |
| CNN | Convolutional Neural Networks |
| MLP | Multi Layered Perceptron |

# CHAPTER 1: INTRODUCTION

## 1.1 Motivation and Problem Statement

The problem of 3D Scene Reconstruction is one of the important challenges in the field of computational photography and computer vision. The goal of this problem is to derive useful geometric information about the environment around us. Modeling 3D environments has many applications, such as digital mapping and navigation, virtual tourism, computer animation, gaming, virtual and augmented reality (VR/AR), robot navigation. Some of these applications, such as mapping and navigation, require accurate 3D reconstructions for precise geometry measurements/estimations. Whereas some applications related to VR/AR applications require visually pleasing, complete, and realistic 3D reconstruction to enable a feeling of presence in virtual or augmented 3D worlds.

Many technologies and methods have been proposed to meet the needs of different applications of 3D Reconstruction. Each of these proposed methods come with their own set of strengths and limitations. These methods can roughly be categorized into active (e.g., structured light sensors and time of flight sensors) and passive acquisition (using multi-view images from cameras) techniques. Although active methods, in general, provide better scene depth information than passive techniques, they usually require external projectors or lasers, which introduces additional power requirements on the 3D capture system. Whereas passive techniques have the advantage of using consumer digital cameras with low power requirements. Moreover, the prevalence and the availability of consumer digital cameras in mobile technology widens the scope of the utility of passive techniques in many day-to-day user applications related to 3D Reconstructions.

Image-based 3D reconstruction works on the principle of the human visual system where like our eyes, using two or more images, the scene depth information can be extracted by a process

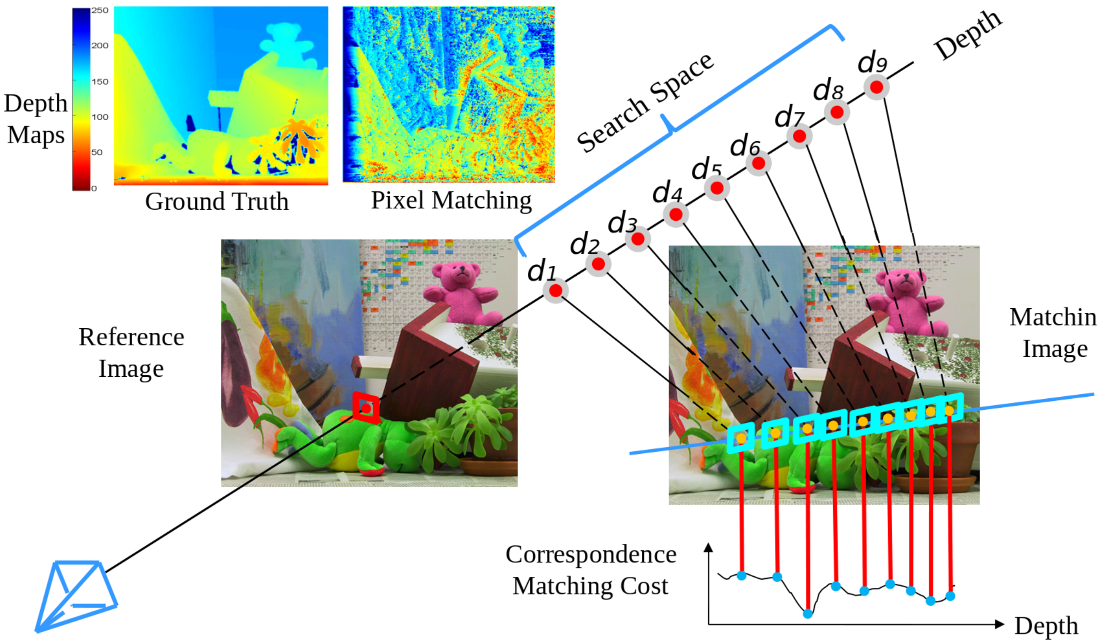


Figure 1.1: A simple example demonstrating a 2D-2D correspondence example that can help to retrieve depth information of the scene. In this example, the pixel similarity metric (absolute difference between pixel intensities) is used as a simple correspondence function. By plotting this similarity metric cost vs. increasing depth, an optimal depth value could be found that corresponds to the local minimum.

known as multi-view stereopsis. After decades of research in the area, the standard algorithm has been set to estimating depth information with help of 2D-2D image correspondences, as shown in Fig 1.1. While the standard mechanism seems very simple, solutions to many underlying problems are still not well established. For example, the optimal method for describing a good 2D-2D correspondence function is still a challenging problem. The success of the correspondence function relies on the robustness and uniqueness of the local image descriptors. While it is simple to obtain a strong descriptor in the regions with well-defined textures, but it is difficult to obtain strong and unique descriptions for the regions with homogeneous texture.

The standard robust feature descriptors such as SIFT (Lowe, 2004) tend to be *sparse* and well defined only in high texture regions. Although these *sparse* descriptors provide sufficient information for many applications, including Simultaneous Localization and Mapping(SLAM), image registration and camera calibration but dense 3D reconstruction requires denser feature descriptors. Several machine learning methods have been introduced that provide very robust

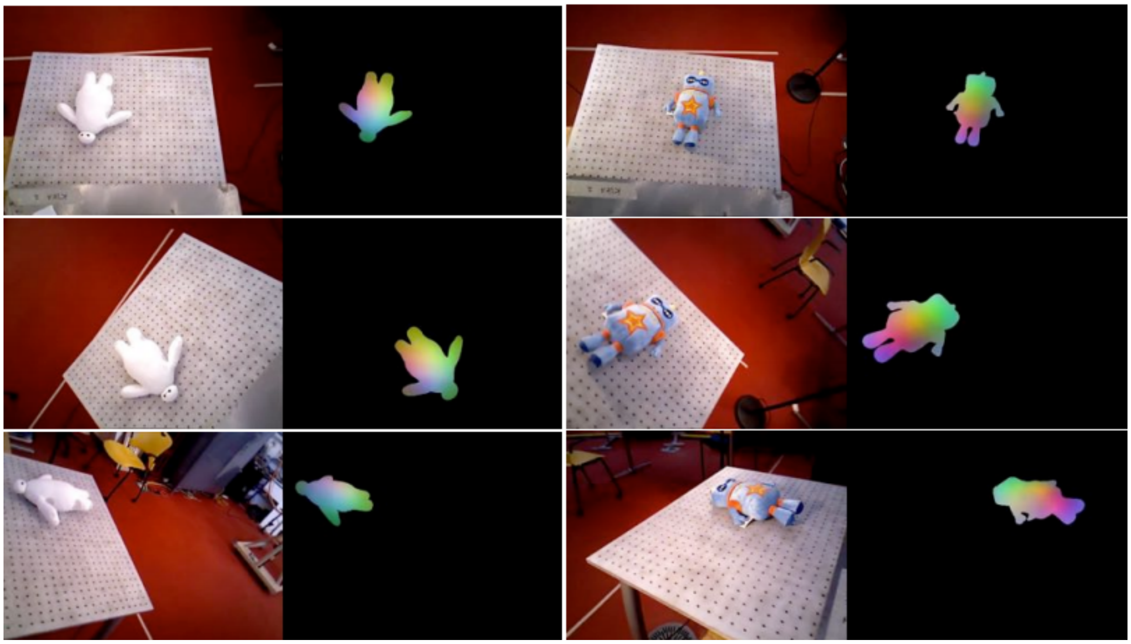


Figure 1.2: An example from (Florence et al., 2018) visualizing the dense local features in RGB color map for two soft toys in various poses. These dense features can be used for making dense 2D-2D correspondences in multi-views.

*dense* image descriptors (Schmidt et al., 2016; Florence et al., 2018) as shown in Fig 1.2. These methods use deep convolution learning framework to automatically learn robust and more general feature descriptors with the help of known ground truth geometry of several shapes during the training process. Unlike standard handcrafted image descriptors, these learned descriptors utilize both local image features and global shape information. The addition of global shape information to an image descriptor makes them robust, unique, and ideal for making dense correspondences. The learning methods can be trained to learn descriptors for obtaining correspondences in challenging scenarios such as specular surfaces and regions with shadows. We discuss this in detail in Chapter  3.

As the standard 2D-2D image correspondences method involves predicting depth information for every pixel individually, ensuring global depth consistency for the entire image is a hard problem. In the literature, many methods have been proposed that utilize local neighborhood depth consistency (either first order or second order) for every image pixel in a global optimization scheme. It is a challenging task to decide the nature of this local neighborhood function and how big ideally this local neighborhood should be. In general, the traditional methods in the literature face problems defining local depth consistencies near depth discontinuities such as corners and boundaries of the objects and thin structures like plants. Recent techniques in the literature (Kendall et al., 2017; Chang and Chen, 2018) have tried to solve the global depth consistency using a deep learning framework. By using learned convolution filters, the hard problem of obtaining a general and robust local neighborhood depth consistency model can be solved. Priors over local neighborhood consistencies can be learned using examples of several kinds of shapes in the training dataset with varying thicknesses and sizes. In such frameworks, the global scene context and semantic scene information can also be utilized. For example, the framework can be trained to recognize pixels of an outdoor image belonging to the sky to be always at infinite depth. Similarly, such systems can be trained to learn priors such as walls of most of the indoor scenes are planar. This process of learning depth cues from the scene context appears to be similar to the human depth perception system. For example, humans can perceive approximate scene depth with a single eye or for a single image using their knowledge of the average sizes of individual objects and the perspective distortion. Similarly, a machine learning framework can be trained to learn these depth cues using a variety of examples scenes during the training process.

In many applications, a depth map from a single viewpoint may not be enough to prepare a 3D reconstruction of the entire scene or an environment. A collection of depth maps from several viewpoints should be fused together to prepare a consistent and complete 3d scene reconstruction. The simplest 3D reconstruction representation is a collection of 3D points gathered from several depth maps. In the literature, this representation is also known as *3D point cloud* representation. While this representation provides accurate 3D points, it lacks surface information, which could be useful for many applications of 3D reconstruction. In the literature, many methods have been discussed to prepare 3D surfaces from unstructured point clouds. Most of these methods try to formulate an *implicit surface* representation, which can be thought of as a function $F(p)$:

$$F(p) = 0, p \in R^3 \tag{1.1}$$

, where $p$ refers to any spatial point. One of the very special properties of this *implicit function* is that it is continuous, which in theory, can provide arbitrary surface resolution and has a property of completing partially observed shapes.

In general, obtaining the true *implicit function* for an entire 3D scene is a tough problem. Moreover, the depth samples obtained form multi-view stereo algorithms are prone to have several kinds of noise and outliers. To make this problem tractable, the proposed methods make several assumptions regarding the discretization of the space, noise, or uncertainty in the input and the nature of the true surface geometry. For example, a popular method for 3D surface reconstruction (Kazhdan et al., 2006) tries to fit a global implicit function to the input set of points. This method is shown to work very well on clean or synthetic data, but in the presence of noise, outliers, and incomplete data, the global surface fitting is prone to produce several artifacts as no prior information is used to predict watertight surfaces in such non-trivial cases. Scaling such global surface fitting to large scenes also remains a challenge.

Whereas, local volumetric integration methods such as (Curless and Levoy, 1996) are more robust to noise in the data as these methods make a scalar approximation of the true *implicit surface* on very small cells of a regular grid. However, these methods cannot make use of the shape completion property of the *implicit surface* representation. Moreover, such algorithms use fixed parameters and functions to describe the region of uncertainties and fusion weights for input samples. While this technique provides reasonable practical solutions in most cases, fixed and heuristic functions are still prone to errors in some non-trivial cases. For example, such methods fail to represent thin structures even at very high volumetric resolutions as the reliance on fixed parameters and heuristic functions makes it hard to resolve both front and backside of such thin surfaces in the presence of noise in the input samples.

Many applications, particularly those running on mobile systems, have memory limitations for the storage of internal *implicit surface* representation for the scenes. As the scenes and environments become bigger and bigger, the size of the many proposed internal *implicit surface* representation methods reach memory limitations. Therefore, in general, the internal *implicit surface* representation is expected to be memory efficient.

Learning *implicit surface* representation from the available ground truth shapes is an interesting solution to the problems discussed above. In recent literature, some methods have been proposed that learn the common statistics and properties of the classes of simple shapes such as chairs, sofas. These methods take advantage of associating both the global and local shape properties to produce globally consistent 3D scene reconstructions. Majority of these methods predict reconstructions using discrete representations such as *point clouds*, *meshes* and *voxels*. Like their classical counterparts, these methods have problems related to their underlying surface representation. Very few and very recent methods, such as (Park et al., 2019), have tried to predict *implicit surface* representation with the help of a deep learning framework. The deep learning framework learns to separately capture both the mean and variances of the training shapes into separate network parameters. In the inference process, the framework utilizes the learned mean or common properties of the trained shapes and optimizes just the variance parameters of the input shape to fit the resultant shape to the input observations. As such methods try to learn and optimize true *implicit surface* representation, they preserve all of its properties, including shape completion and arbitrary surface resolution without the use of any enforced assumptions and approximation, unlike many classical approaches. We illustrate such properties in Fig 1.3.

Although, it is still challenging to express large scenes with a single true *implicit surface* representation. The authors of the work DeepSDF (Park et al., 2019) have shown the learned *implicit surface* representation of very small and simple objects such as chairs, tables, lamps, etc. Whether such representations can be extended to large scenes efficiently still remains an important and interesting question. In this dissertation (Chapter. 5), we try to answer this question with a simple but promising solution.

In this thesis, we investigate a machine learning framework to solve several problems related to 3D reconstruction using image-based methods. This thesis concentrates on two aspects of 3D reconstruction.

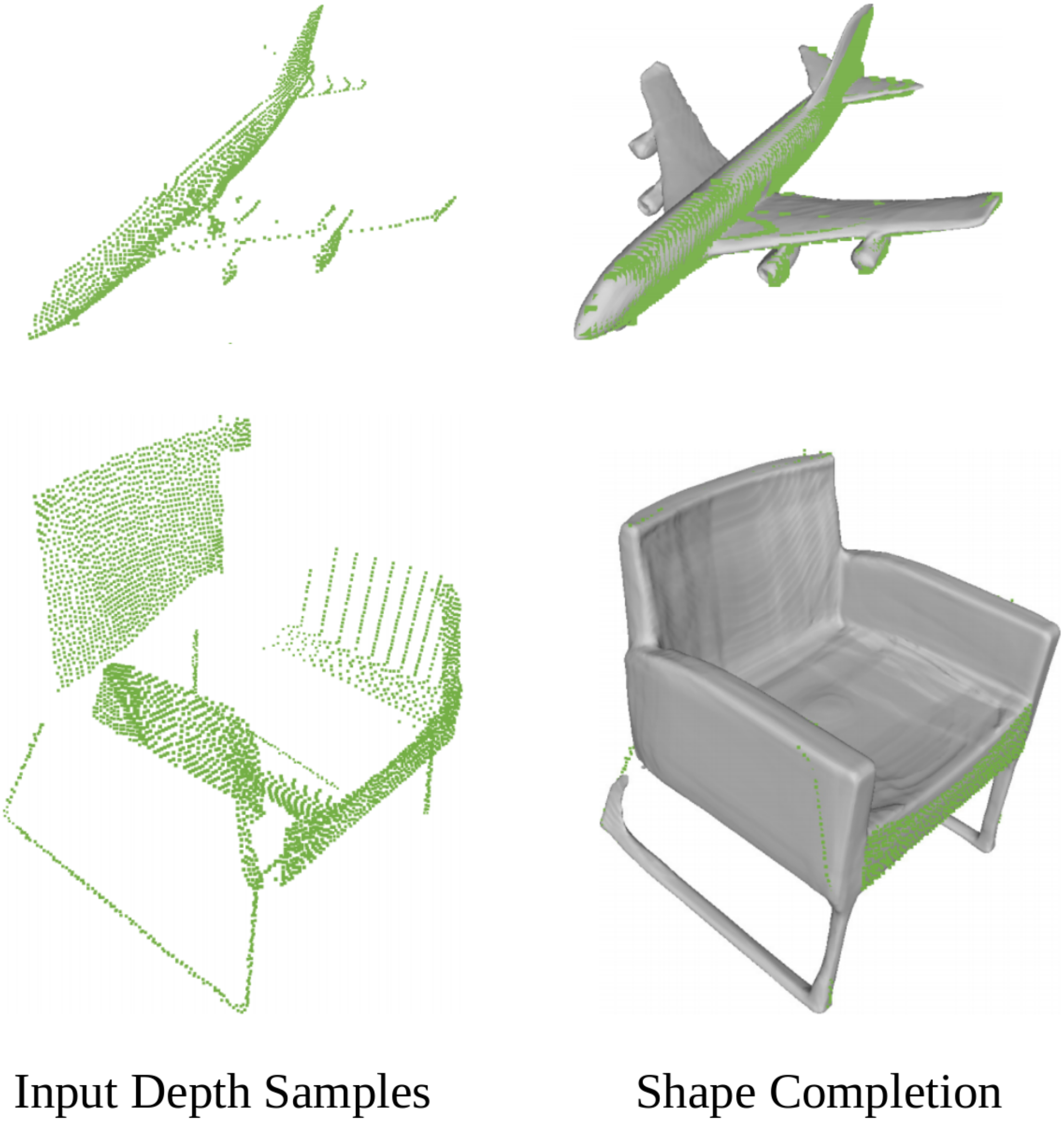


Figure 1.3: Examples of shape competion taken from DeepSDF (Park et al., 2019). These examples show the shape completion property of learned *implicit surface* representation.

1. *Depth estimation and 3D scene reconstruction from the input stereo images*. While the prior work in machine learning has taken long strides in solving many hard problems related to image-based 3D reconstruction and depth estimation, but still many underlying problems need to be solved. For example, many methods in deep learning literature have been shown to produce reasonable depth estimation from stereo images, but their depth maps are usually not geometrically consistent enough to be fused into a consistent 3D reconstruction. Moreover, often, the depth estimation of occluded regions is prone to be noisy and needs to be filtered correctly before the fusion step. To tackle such problems, in this thesis, we learn to produce occlusion aware and geometrically consistent depth estimation from stereo images. Furthermore, we show that these depth images can be used to produce 3D reconstruction of challenging indoor scenes.

2. *Learning local shape priors for detailed 3D reconstruction*. While state-of-the-art learning *implicit surface* representation methods provide shape completion properties, but they have only been shown to learn simple classes of shapes such as chairs and sofas. It usually takes several hours to optimize these simple shapes with about millions of parameters. To learn and optimize 3D environments and natural 3D scenes, these methods might take an impractical amount of time and parameters (several days to learn a shape such as Stanford Bunny). To tackle this problem in this thesis, we instead learn priors on local shapes in the regular 3D grids and enforce the local neighborhood consistencies on them to achieve globally consistent surfaces. In contrast to the previously proposed global shape representation learning techniques our approach is more efficient as the space of possible local shapes is much smaller than of general shapes and scenes, the learning process is simple and hits very early convergence with orders of magnitude smaller number of parameters. As our proposed method is local, it is composed of several independent local *implicit functions* hence it does not provides global shape completion. In this thesis, we illustrate that using deep local shape priors; we can represent the general 3D scene environments both more accurately and more efficiently than the previous state-of-the-art 3D reconstruction meth-

ods. Moreover, the resultant scene representation is still an *implicit function* and preserves it's properties, including shape completion and shape generation but in the scope of the physical extent of these local shapes.

## 1.2 Related Work

This section starts with a summary of prior work on systems for building 3D Scene Reconstruction using traditional methods. Next, a detailed summary of prior work is provided on methods that use machine learning for image-based 3D reconstruction. In the later section, several methods, including learning-based depth estimation from passive images and some very recent 3D scene representation methods, are discussed.

### 1.2.1 3D Reconstruction using Traditional Methods

This section starts with discussing methods for obtaining depth information from general scenes as this is the most vital requirement for any reconstruction system. Next, several algorithms used in the literature for 3D scene representation are discussed.

#### 1.2.1.1 Depth Estimation

Scene depth information can either be obtained from depth/range sensors or by using passive stereo methods.

**Active Methods**:- Traditionally, depth sensors work on the principle of Structured Light or Time of Flight. Both these technologies are limited to work indoors and fail to work in sunlight conditions. The range scanners such as LIDAR, etc. have low resolution and could not be used in AR/VR based applications due to limitations in their size and resources they require. More information on depth sensors can be found in survey (Zollhöfer et al., 2018).

**Passive Methods**:- Depth from stereo has been widely explored in the literature; we refer interested readers to surveys and methods described in (Scharstein and Szeliski, 2002). There has been a lot of work in multi-view stereo based reconstruction where depth for a reference view is obtained from many matching views. While the basic concept remains similar to two-view stereo but having many views does not only increases the stereo search complexity but also raises several questions related to view selection, robustness to occlusions, and geometric consistencies over multiple views. A deep survey and comparisons of classical MVS based reconstruction methods can be found in (Seitz et al., 2006). There has been a lot of work on real-time reconstruction using passive camera motion, often known as Dense Visual SLAM (Newcombe, 2012). These methods try to optimize for camera motion simultaneously and surface geometry (Newcombe et al., 2011b; Pradeep et al., 2013; Ummenhofer et al., 2017).

#### 1.2.1.2 3D Scene Representation Methods

3D reconstruction using depth information has been a widely studied topic in computer vision. The detailed state-of-the-art review can be found in the survey (Zollhöfer et al., 2018).

The reconstruction methods could be roughly categorized into four categories. Voxel-based methods (Curless and Levoy, 1996; Klein and Murray, 2007; Stühmer et al., 2010; Newcombe and Davison, 2010; Newcombe et al., 2011a), Points or Surfels based methods (Pfister et al., 2000; Ummenhofer and Brox, 2013; Keller et al., 2013), Global Implicit Function approximation methods (Carr et al., 2001; Kazhdan et al., 2006; Fuhrmann and Goesele, 2014) and Visibility or Free Space Constraints based methods (Labatut et al., 2009; Jancosek and Pajdla, 2011; Aroudj et al., 2017).

### 1.2.2 3D Reconstruction using Learning Based Methods

This section discusses in detail the depth from learning-based stereo methods and learning-based 3D Shape/Scene representation methods.

#### 1.2.2.1 Learning Depth Estimation

In recent years, there has been significant improvement in the quality of depth estimation from stereo images using machine learning techniques such as Convolution Neural Networks. The depth maps produced by these techniques achieve much higher robustness and completion than the previously proposed traditional methods.

The first work in this body of research, MC-CNN (Zbontar et al., 2016), proposed a siamese neural network (Chopra et al., 2005) to compare two image patches, where the network used the same weights while working in tandem on two different input image patches and produced a real valued correlation between them. This learning method enabled the learning of robust stereo cross-correlation metrics from data, which was plugged into a classical semi-global matching process (Hirschmuller, 2008) to predict consistent disparity estimation. DispNet (Mayer et al., 2016) improved the previous method by using an end-to-end disparity estimation deep neural network with a correlation layer (dot product of features) for stereo volume construction. This method enabled learning priors over the global scene context and not just the stereo-cost metric. GC-Net (Kendall et al., 2017) improved the previous method by using high dimensional feature vector in building stereo cost volume instead of using the dot product of feature vectors as used in previous work. PSMNet (Chang and Chen, 2018) improved GC-Net by enriching extracted features from CNN with a better global context using a pyramid spatial pooling process. They also showed the effective use of residual learning networks in the cost filtering process.

Several methods such as CRL (Pang et al., 2017), iResNet (Liang et al., 2018), StereoNet (Khamis et al., 2018) and FlowNet2 (Ilg et al., 2017) proposed depth refinement using residual learning with guidance of photo-metric error (either in image or feature domain).

#### 1.2.2.2 3D Scene Representation Learning

Recently there has been rapid progress in 3D Shape Learning using deep neural networks. Point-based methods such as PointNet (Qi et al., 2017) try to learn global shape features from a set of input points and use encoder based generative models for scene representation (Achlioptas et al., 2017; Yang et al., 2017). Mesh-based methods either use existing (Sinha et al., 2016) or learned (Groueix et al., 2018; Ben-Hamu et al., 2018) parameterization techniques to describe 3D surfaces by morphing 2D planes. Works such as (Groueix et al., 2018; Ben-Hamu et al., 2018) use sphere parameterization to produce the closed mesh. The voxel-based method does not usually preserve fine shape details (Wu et al., 2015; Choy et al., 2016). Octree-based methods (Tatarchenko et al., 2017; Riegler et al., 2017; Häne et al., 2017) alleviate the compute and memory limitations of dense voxel methods, extending the voxel resolution up to $512^3$. Aside from occupancy grids, a class of work (Dai et al., 2017; Zeng et al., 2017; Stutz and Geiger, 2018) extract local shape geometric descriptors from signed distance functions of local grids (Input is SDF and output is descriptor). Unlike these methods, we extract local shape descriptors directly from depth samples and approximate globally consistent SDF for the entire shape/scene.

More relevant to my research are the methods that approximate global implicit functions. Occupancy Networks (Mescheder et al., 2019) tries to approximate shapes using occupancy-based implicit function; similarly, DeepSDF (Park et al., 2019) approximates shapes using Signed Distance Fields. Fundamentally both methods use continuous surface representation, but signed distance fields promise better resolution with minimum computation effort. Hence we adopt DeepSDF as the backbone architecture for my work on scene reconstruction using local shape priors.

### 1.3 Thesis Contributions

1. *Deep learning-based Stereo Vision*: We propose a novel depth learning system to produce geometrically consistent depth and occlusion maps from stereo images that can be used in fusion systems such as KinectFusion (Newcombe et al., 2011a) to produce high quality 3D reconstructions from passive image data. We introduce a neural architectural improvement over existing 3D cost filtering methods by the use of configurable receptive fields. Our proposed cost filtering method achieves significant improvement in both accuracy and speed over the state-of-the-art methods. We also propose a method that preserves thin and sharp

corners for depth prediction while transferring learning from simulation to real scenes. Furthermore, we demonstrate that my proposed method produces consistent 3D reconstructions even in very challenging scenes with large texture-less walls, thin structures, regions with specular highlights, and dark shadows. This work on learning geometrically consistent depth estimation from stereo images and subsequent fusion of the learned depth into a 3D scene reconstruction was published as StereoDRNet (Chabra et al., 2019).

2. *Deep Local Shape Priors*: We propose a novel method that uses a deep learning system to learn a generalized *implicit surface* representation as Signed Distance Functions (SDF) for natural 3D scenes and environments from the available noisy scene depth information. In contrast to the prior work that learns SDF for template shapes, the proposed method learns to produce SDF over local regions making the overall system more efficient, accurate, and scalable to general scenes. In the thesis, we illustrate that the local neighbor consistency regularization and overlapping receptive field of local shapes help to optimize consistent 3D reconstruction of the entire scene. The proposed system produces reconstructions with orders of magnitude improvement in accuracy and network compression over state-of-the-art learning-based 3d shape representation methods. The proposed method outperforms the existing approaches in dense 3D reconstruction from partial observations, showing thin details with significantly better surface completion and high compression. The technical report on learning these local shape priors from detailed 3D Reconstruction was published as DeepLS (Chabra et al., 2020).

# CHAPTER 2: TECHNICAL INTRODUCTION

## 2.1 Deep Learning for 3D Vision

The extraction of meaningful information from three-dimensional (3D) sensed data is a fundamental challenge in the field of computer vision. Much like with two-dimensional (2D) image understanding, 3D understanding has greatly benefited from the current technological surge in the field of machine learning. With applications ranging from depth sensing, mapping and localization, autonomous-driving, virtual/augmented reality, and robotics, it is clear why learning of robust functions and representation models from 3D data is in high demand. Currently, both academic and industrial organizations are undertaking extensive research to explore this very active field further.

Classic machine learning methods such as Support Vector Machines (SVM) and Random Forest (RF) have typically relied on a range of handcrafted shape descriptors (i.e., Local Surface Patches (Chen and Bhanu, 2007), Intrinsic Shape Signatures (Zhong, 2009), Heat Kernel Signatures (Sun et al., 2009), etc.) as feature vectors from which to learn. These methods have delivered successful results in a range of 3D object categorization and recognition tasks. However, much like in the field of 2D image understanding, there has been a shift in focus to a deep learning approach (LeCun et al., 2015).

Deep learning approaches differ from other machine learning approaches in that features themselves are learned as part of the training process. This process is commonly referred to as representation learning, where raw data is passed into the learning algorithm, and the representations required for detection or classification are automatically derived using the process known as back-propagation (Hecht-Nielsen, 1992). This ability to learn features is often seen as the cause for the rapid improvement in 3D understanding benchmark results, including depth estimation and 3D reconstruction tasks.

In this chapter, we first discuss the basics of deep learning, different types of neural networks, activation functions, and common neural network architectures. Advancement in learning depth estimation and 3D representation learning are also discussed.

### 2.1.1 Deep Artificial Neural Networks

Deep Learning is a set of learning methods that attempt to model data with complex architectures combining several non-linear functions. The elementary building blocks to deep learning systems are artificial neural networks that are combined together several times to yield deep neural networks.

An artificial neural network defines a non-linear function model $f$ with respect to the model parameters $\theta$ (priors) and the input $x$ and the function's output $y$ can be mathematically written as $y = f(x, \theta)$ . These networks can be used to learn classification or regression tasks. These model parameters $\theta$ can be estimated from learning samples. The success of such functions was proposed in universal approximation theorem (Hornik et al., 1989). Furthermore, (LeCun et al., 1989) introduced an efficient way to compute the gradient of neural networks using a back-propagation algorithm.

#### 2.1.1.1 Artificial Neuron

An artificial neuron is a function $f_j$ of the input $x = (x_1, ..., x_d)$ weighted by a vector of connection weights $w_j = (w_{j1}, ..., w_{jd})$, completed by a neuron bias $b_j$ , and associated to an activation function $\phi$, namely

$$y_j = f_j(x) = \phi(\sum_{k=0}^{d} w_{jk} x_j + b_j) \tag{2.1}$$

Several activation functions can be considered.

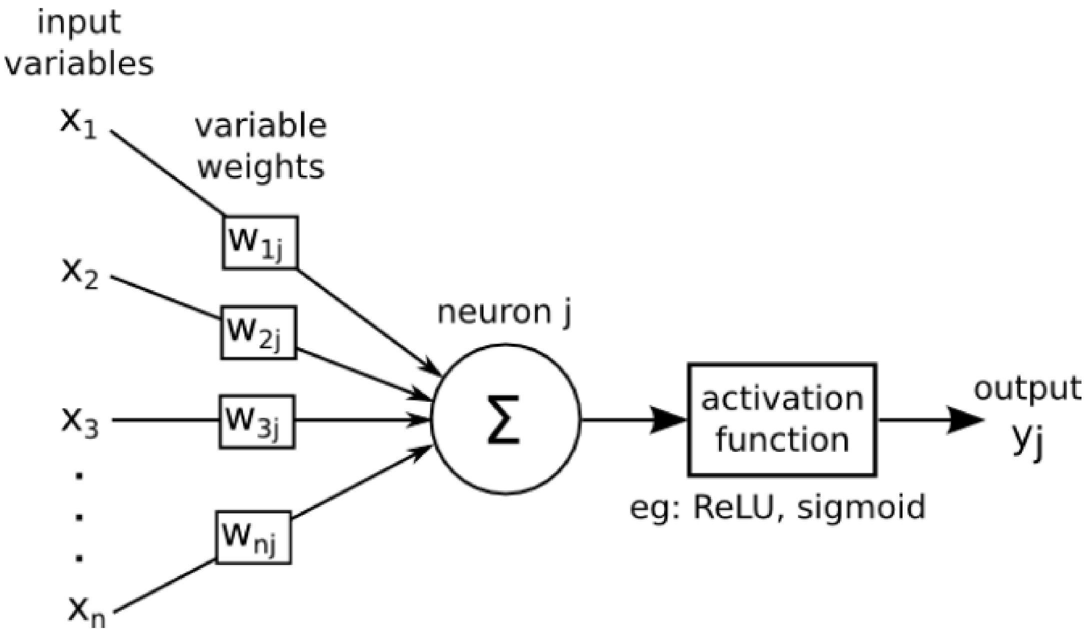


Figure 2.1: An illustration of a typical Artificial Neuron

- The identity function

$$\phi(x) = 1 \tag{2.2}$$

- The sigmoid function

$$\phi(x) = \frac{1}{1+e^{-x}} \tag{2.3}$$

- The hyperbolic tangent function(tanh)

$$\phi(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.4}$$

- The Rectified Linear Unit(ReLU) function

$$\phi(x) = max(0, x) \tag{2.5}$$

Fig 2.1 shows a typical artificial neuron, and Fig 2.2 shows the mathematical nature of some activation functions.

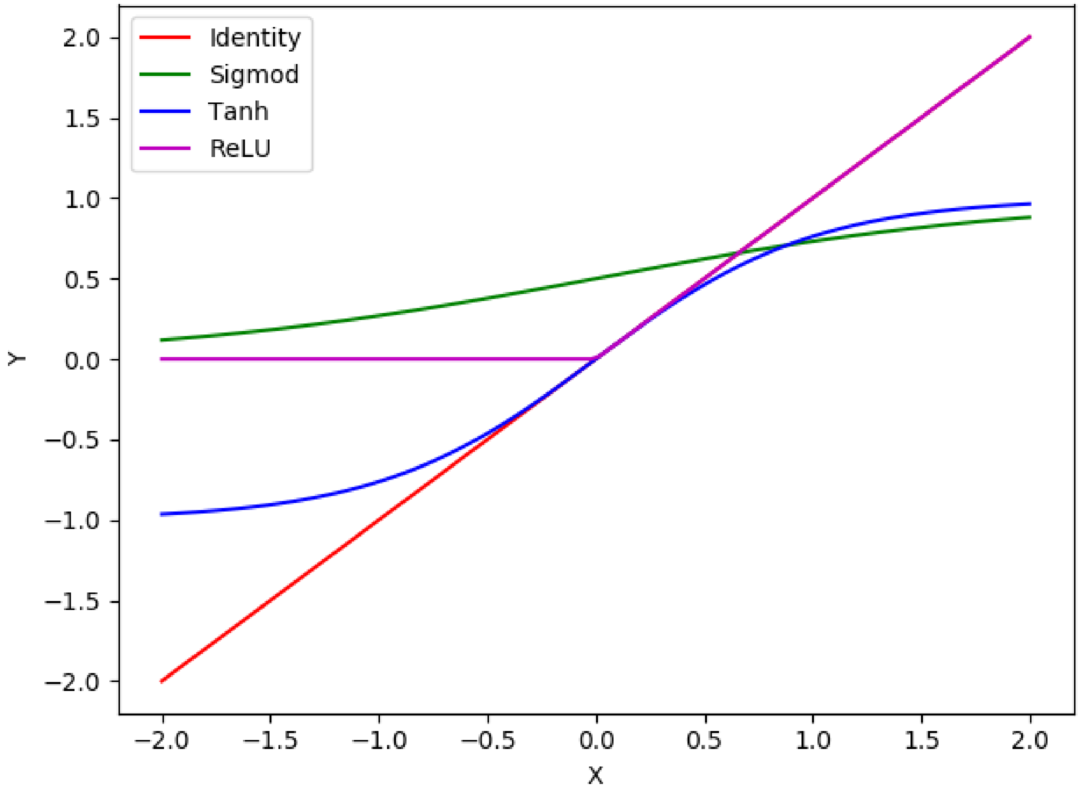


Figure 2.2: Activation Functions

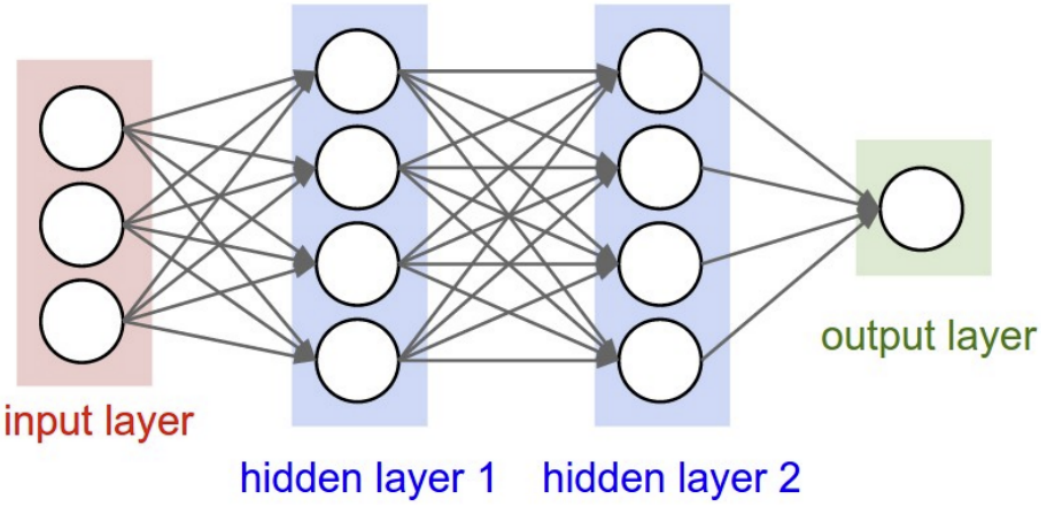


Figure 2.3: A simple Multi Layer Perceptron

#### 2.1.1.2 Multi Layer Perceptrons (MLP)

A multilayer perceptron is a structure composed of several hidden layers of neurons where the output of a neuron of a layer becomes the input of a neuron of the next layer. On the output of last layer of MLP various activation functions can be applied. The choice of the activation depends the type of tasks to be solved by the neural network architecture. For regression tasks typically identity or ReLU activation functions are used and for classification tasks the choice of sigmoid activation function is generally considered.

Multilayer perceptrons are fully connected networks, i.e., each unit (or neuron) of a layer is linked to all the neurons of the next layer. The hyper-parameters of the architecture are the number of hidden layers and the number of neurons in each layer. An example of simple two-layered MLP is shown in Fig 2.3

#### 2.1.1.3 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a special type of multilayer perceptron neural network. Unlike typical MLPs, the neurons of a layer of CNN are not linked with all neurons of the next layer. In fact, as their name indicates, CNN's utilize convolution operations to introduce the local connectivity of the neurons. The parameters $\theta$ of these networks are smaller, local, and shared. Thus, making them powerful to train than a typical MLP. Although, CNNs can only be used when the input data is a structured grid, they have revolutionized image processing especially in tasks of learning robust features on natural images (ImageNet (Krizhevsky et al., 2012)).

A Convolutional Neural Network can be composed of several types of layers, namely convolution layer, pooling layer, or fully connected layer. The discrete convolution between two functions $f$ and $g$ is defined as

$$(f * g)(x) = \sum_{t} f(t)g(x+t) \tag{2.6}$$

For 2-dimensional signals such as images, we consider the 2D-convolutions

$$(K * I)(i,j) = \sum_{m,n} K(m,n)I(i+n, j+m) \tag{2.7}$$

$K$ is a convolution kernel applied to a 2D signal or an image $I$. The principle of 2D convolution is to drag a convolution kernel $K$ on the image. In *convolution layers*, the kernels can be used to skip pixels defined by a number $s$, known as a stride. In each layer of CNN, zero padding can be used to control the size of the output of the layer.

CNN also has pooling layers, which allow reducing the dimension, also referred to as subsampling, by taking the mean or the maximum on patches of the image (mean-pooling or max-pooling). Like the convolutional layers, pooling layers acts on small patches of the image; we also have a stride. If we consider 2 × 2 patches, over which we take the maximum value to define the output layer, and a stride s = 2, we divide by 2 the width and height of the image. An example of a famous CNN is shown in Fig 2.4.

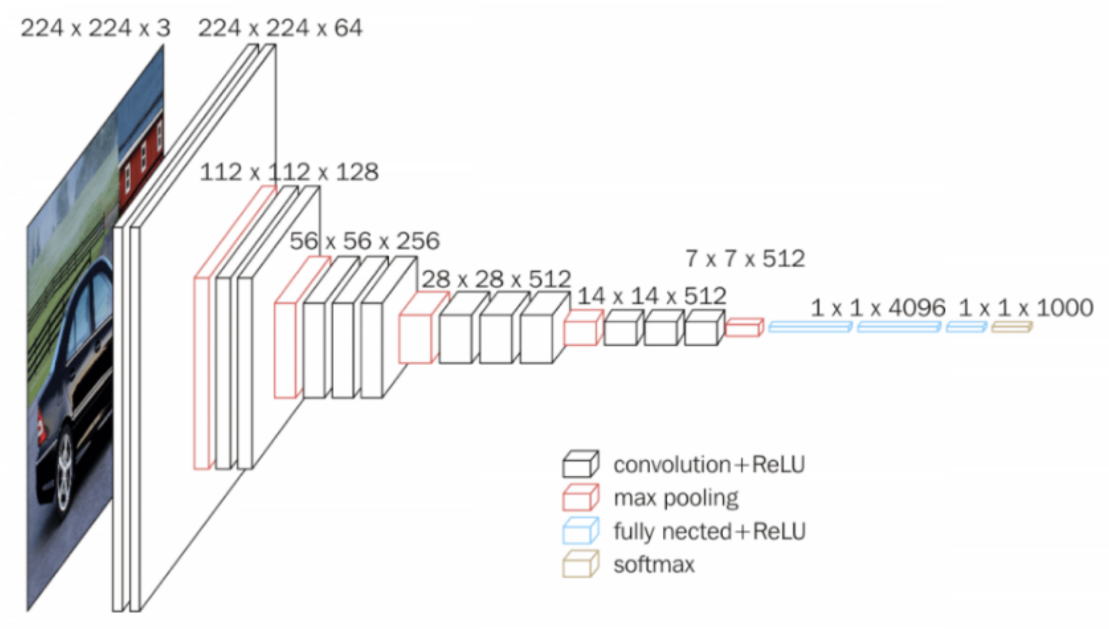


Figure 2.4: VGG16, a convolutional neural network model proposed by  (Krizhevsky et al., 2012) for large scale image classification tasks.

#### 2.1.1.4 Optimization Algorithms

The goal is to find an optimal mapping function $f(x)$ to minimize the loss function of the training samples

$$\min_{\theta} \frac{1}{N} \sum_{j=1}^{N} L(y_j, f(x_j, \theta)) \tag{2.8}$$

where $N$ is the number of training samples, $\theta$ are the parameters (the weights $w_j$ and biases $b_j$) of the mapping function, $x_j$ is the feature vector of the jth input sample, $y_j$ is the corresponding label, and $L$ is the loss function. In the field of machine learning, the most commonly used optimization methods are mainly based on gradient descent (Ruder, 2016). The idea of the gradient descent method is that variables update iteratively in the (opposite) direction of the gradients of the objective function. The update is performed to gradually converge to the optimal value of the objective function. The learning rate $\eta$ determines the step size in each iteration, and thus influences the number of iterations to reach the optimal value. Since the batch gradient descent has high computational complexity in each iteration for large-scale data and does not allow online update, stochastic gradient descent (SGD) was proposed (Robbins and Monro, 1951). The idea of stochastic gradient descent is using one sample randomly to update the gradient per iteration, instead of directly calculating the exact value of the gradient. For detailed information on optimization techniques related to machine learning we encourage readers to refer the survey (Sun et al., 2019).

#### 2.1.1.5 Loss Functions

Once the architecture of the network has been chosen, the parameters $\theta$ (the weights $w_j$ and biases $b_j$ ) have to be estimated from a learning sample. As usual, the estimation is obtained by minimizing a loss function with a gradient descent algorithm. Broadly, loss functions can be classified into two major categories depending upon the learning task we are dealing with — Regression losses and Classification losses. In classification, we are trying to predict the output from a set of finite categorical values i.e. Categorizing animals into different categories from a large data set of images. Regression, on the other hand, deals with predicting real-valued functions, for example, estimating scene depth, predicting prices of stocks, etc. We first discuss loss functions that are used for classification tasks.

- **Binary Cross-Entropy Loss (BCE)**:- This is the most common setting for classification problems. Cross-entropy loss increases as the predicted probability diverge from the actual label. This loss function can be written as:-

  $$L = (Y)(-log(\hat{Y})) + (1-Y)(-log(1-\hat{Y}) \tag{2.9}$$

  Where, $Y$ is the ground truth label and $\hat{Y}$ is the predicted label.

- **Multi Class Classification using Softmax**:- Multiclass classification is appropriate when we need our model to predict one possible class output every time. In order to extend BCE to multiclass classification the softmax operator is first used followed by sum of log loss for each category. We define Softmax function $p_i$

$$p_i = \frac{e^{y_i}}{\sum_{i=0}^{n} e^{y_i}} \tag{2.10}$$

The loss function is defined as

$$L = \sum_{i=0}^{n} -log(p_i) \tag{2.11}$$

We now discuss several commonly used regression loss functions.

- **Mean Absolute Error (L1 Loss)**:- As the name suggests, mean absolute error is measured as the average of the difference between predictions and actual observations. We define MAE loss as

  $$L = \frac{\sum_{i=0}^{n} |y_i - \hat{y_i}|}{n} \tag{2.12}$$

  Where, $y_i$ is the ground truth value and $\hat{y_i}$ is the predicted value.

- **Mean Square Error (L2 Loss)**:- As the name suggests, mean square error is measured as the average of squared difference between predictions and actual observations. We define MSE loss as

  $$L = \frac{\sum_{i=0}^{n} (y_i - \hat{y_i})^2}{n} \tag{2.13}$$

  Where, $y_i$ is the ground truth value and $\hat{y_i}$ is the predicted value.

- **Smooth L1 Loss (Huber Loss)**:- Smooth L1-loss can be interpreted as a combination of L1-loss and L2-loss. It behaves as L1-loss when the absolute value of the argument is high, and it behaves like L2-loss when the absolute value of the argument is close to zero. The huber loss $H_x$ equation is:

  $$H(x) = \begin{cases} |x|, & \text{if } |x| > \alpha \\ \frac{x^2}{|\alpha|}, & \text{otherwise} \end{cases} \tag{2.14}$$

Where, $\alpha$ is a hyper-parameter here and is usually taken as 1. The smooth L1 loss function can now be written as

$$L = \frac{\sum_{i=0}^{n} H(|y_i - \hat{y_i}|)}{n} \tag{2.15}$$

Where $y_i$ is the ground truth value, and $\hat{y_i}$ is the predicted value.

### 2.1.2 Datasets for 3D Learning

High-quality data play a central role in enabling many computer vision technologies. For example, the evolution of stereo algorithms greatly benefited from the Middlebury dataset (Scharstein and Szeliski, 2003). However, the collection and labeling process of large-scale datasets for 3D vision tasks are tedious and costly. KITTI (Geiger et al., 2012); DTU (Jensen et al., 2014), and ETH3D (Schops et al., 2017) all require carefully calibrated camera rigs to capture images, expensive Laser scanner or structured light sensor to obtain the ground truth.

One popular way to collect training data is to utilize photo-realistic rendering engines to synthesize large amounts of data with known ground-truth. Models trained on such synthetic datasets demonstrated on-par or even better performances with respect to models trained on real-world datasets. For example, SceneFlow (Mayer et al., 2016) employed rendering software to generate data for scene flow estimation tasks.

In the recent past, there has been a significant attempt to release real scene datasets for large scale indoor scene reconstruction, namely Matterport (Chang et al., 2017) and highly photo-realistic dataset Replica (Straub et al., 2019b). Although such real scene datasets usually do not provide ground truth geometry. Synthetic large scale scene datasets such as ICL-NUIM (Handa et al., 2014) is often used as a benchmark for 3D reconstruction systems. Whereas, shape representation methods often use ShapeNet (Chang et al., 2015) dataset. This dataset provides a large variety of different classes of objects such as sofas, chairs, etc.

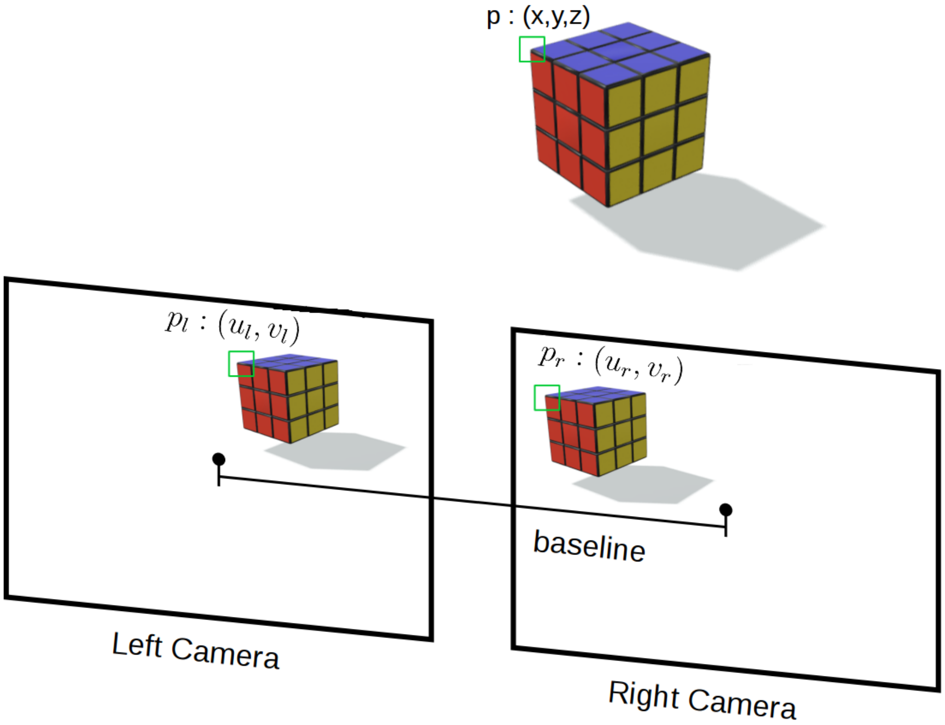


Figure 2.5: An example showing stereo views for an object. The reference corner point $p$ (shown by green marker) of the object is viewed at different pixel locations $p_l$ and $p_r$ in left and right cameras respectively.

## 2.2 Depth From Stereo Vision

In stereo vision, the two cameras are displaced from one another (usually horizontally) by a fixed distance known as *baseline*. This relative shift in camera geometry introduces two different views of a scene. Due to this difference in the views of the cameras, the features captured in the images have different locations in the stereo-images, as shown in Fig 2.6. The identification of the difference in the location of the common features in stereo-views is known as stereo matching [(Hartley and Zisserman, 2003)] and produces a pairing $m(p_l, p_r)$ where $p_l$ and $p_r$ are 2D feature locations in the left and right images respectively.

Once the pairing $m(p_l, p_r)$ is obtained by the stereo-matching process, triangulation can be applied to calculate the distance between the captured objects and the stereo-camera. In order to limit the search of pixel-correspondences $m(p_l, p_r)$ along the same horizontal epipolar line the stereo image pairs can be rectified to epipolar geometry. For detailed description of the process of the stereo camera rectification, the reader is referred to (Hartley and Zisserman, 2003). For rectified stereo-images, disparity $d$ is the difference in the horizontal location between pixels $p_l$ and $p_r$ for the match $m(p_l, p_r)$. Pixels in the left and right image could be represented in image coordinates as $p(u_l, v_l)$ and $p(u_r, v_r)$ respectively, the disparity $d$ can be expressed as $d = u_l - u_r$. An image $D(u_l, v_l)$ where intensity values correspond to disparity $d$ for each match $m(p_l, p_r)$ on the left image is known as disparity map. Disparity $d$ can easily be used to estimate depth $z$ using a simple relation described in (Hartley and Zisserman, 2003) as

$$z = f\frac{b}{d} \tag{2.16}$$

where $b$ and $f$ correspond to stereo camera baseline and focal length respectively. Estimating these parameters of stereo camera is the part of the process known as stereo camera calibration.

### 2.2.1 Stereo Camera Calibration

Stereo camera calibration involves estimating both *intrinsic and extrinsic camera parameters* of the stereo camera pair. *Intrinsic camera parameters* enables practical use of the simple camera model, transforming a 3D point in the frame of reference of the camera into a 2D point in the camera image. *Extrinsic camera parameters* define the relative rigid geometry between two cameras. We will now first discuss the definition of these stereo camera parameters and then describe the calibration process.

- *Intrinsic camera parameters*:- Given a point $P(x, y, z) \in \mathbb{R}^3$ we define perspective projection to a point $u \in \mathbb{R}^2$ as

$$u = \pi(P) \equiv \frac{1}{z}\begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{2.17}$$

where $\pi$ is known as perspective division operator. The intrinsic calibration matrix K is defined as:

$$K \equiv \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \tag{2.18}$$

with focal length $(f_u, f_v)^{\mathsf{T}}$ and principal point $(c_u, c_v)^{\mathsf{T}}$. We can obtain image co-ordinates of a projected point $p$ as

$$p = K\pi(P) \tag{2.19}$$

We back-project the image co-ordinates $p$, back to a point $P$ using a depth $z \in \mathbb{R}$ by:

$$P = (K^{-1}p)z \tag{2.20}$$

- *Extrinsic camera parameters*:- Given a point $P \in \mathbb{R}^3$ in world coordinate system, this point can be represented in left and right camera system as $P_l$ and $P_r$ by

$$P_l = R_l P + t_l \tag{2.21a}$$

$$P_r = R_r P + t_r \tag{2.21b}$$

where $R_l$, $R_r$, $t_l$ and $t_r$ are rotation and translation matrices of left and right cameras. The first Eq. 2.21a can be solved for $P$ and inserted into the second Eq. 2.21b, which gives

$$\begin{aligned} P_r &= R_r R_l^{-1}(P_l - t_l) + t_r \\ &= R_r R_l^{-1}(P_l - t_l + R_l R_r^{-1} t_r) \\ &= R_s(P_l - t_s) \end{aligned} \tag{2.22}$$

where, $R_s$ and $t_s$ are relative rotation and translation matrices between the stereo cameras. They are known as *extrinsic camera parameters* and can be written as

$$R_s = R_r R_l^{-1} \tag{2.23a}$$

$$t_s = t_l - R_l R_r^{-1} t_r \tag{2.23b}$$

We now briefly discuss the calibration process, as described in (Zhang, 2000). In this process, typically, several images of calibration patterns are taken from various viewpoints. The knowledge about the location of calibration points in the real world, and the image permits the calculation of *intrinsic and extrinsic parameters* of a stereo camera pair. Zhang's method requires at least three images of co-planar calibration points from different positions for a unique solution. The analytic calculation uses an algebraic distance to model errors, which is only useful as a first guess. A non-linear optimization step takes this guess and refines the values according to the correct error model. Next, the parameters of radial lens distortion are calculated directly by assuming that all other parameters are constant. Finally, a non-linear optimization takes place using the full model and all parameters.

### 2.2.2 Stereo Matching

Approaches for solving the stereo-matching problem follow a common set of steps in order to perform the matching $m(p_l, p_r)$, where $p_l$ and $p_r$ are 2D feature locations in the left and right camera respectively as shown in Fig 2.6. These steps are feature description, correlation cost calculation, and depth regression. In the following sections, we briefly discuss both the traditional and data-driven methods used for each of these steps.

#### 2.2.2.1 Feature Descriptors

Feature descriptors for an image are a unique set of values (or a vector) that describe useful information. It is important that these features are easy to identify and robust under the view changes produced by the separation of the cameras. Features can be classified according to the information they represent as points, edges or ridges, and segments. Some of them have been found to be robust to view-changes and have been successfully used for stereo-matching (Meltzer and Soatto, 2008; Geiger et al., 2010). In recent work (Zbontar et al., 2016), there has been a paradigm shift to learn these feature descriptors using machine learning approaches. We now briefly discuss some popular handcrafted and machine-learned feature descriptors in the context of stereo matching.

- *Handcrafted descriptors*:- Edge-point-based approaches select an edge-point and use only this point for performing the matching of the edge, assuming there is a direct relationship between the disparity of this edge point and the remaining points in the edge. Edge-point-based approaches  (Meltzer and Soatto, 2008; Geiger et al., 2010; Rao et al., 2012; Wei and Ngan, 2005; Baker, 1982) use either feature points detected along the edge (e.g. corners or end-points) or each of the points independently. (Meltzer and Soatto, 2008) proposed a bio-inspired edge descriptor based on Histograms of Oriented Gradients (HOG) for different scales. The descriptor is created by calculating the HOGs for all of the points in a circle of radius $r$ centered on the anchor point. The orientations are weighted by edge strength and a Gaussian centered at the anchor points. (Fabbri and Kimia, 2010) used tangent attributes on curvelets to describe the image edges. During the matching stage, 3D reprojection information is used additionally to restrict the search space and ensure consistency in the reconstructed objects. Although all of these handcrafted descriptors provide good matching results in regions with high gradient information, they face problems in regions with homogeneous texture data in the images.

- *Learned descriptors*:- Recent methods such as (Zbontar et al., 2016; Mayer et al., 2016; Kendall et al., 2017; Chang and Chen, 2018; Chabra et al., 2019) use deep convolution learning framework to automatically learn robust and more general feature descriptors with the help of known ground truth geometry of several shapes during the training process. Unlike standard handcrafted image descriptors, these learned descriptors utilize both local image features and global shape information. The addition of global shape information to an image descriptor makes them robust, unique and apt for making dense correspondences.

#### 2.2.2.2 Correlation Metrics

Once the descriptors have been created, the dissimilarity $\delta(\xi(p_l), \xi(p_r))$ of a pair of descriptors on the left image and descriptor on the right image must be calculated. These dissimilarities can be measured by correlating the feature descriptors $\xi$ at pixel locations $p_l$ and $p_r$ in the left and right images, respectively. We briefly discuss some of the most popular correlation metrics in the context of stereo matching.

- Absolute Difference (AD) is commonly used as a dissimilarity indicator due to its simplicity. It is defined as:

$$\delta_{AD}(p, d) = |\xi^l(p) - \xi^r(p - d)| \tag{2.24}$$

  where $\xi^l(p)$ is the feature descriptor being used at pixel $p$ in the left image and $\xi^r(p - d)$ is the feature descriptor at corresponding to pixel $p$ at disparity $d$ in the right image.

- Sum of Absolute Difference (SAD) use the sum of absolute difference over intensity neighbourhood descriptor $\xi_W(p)$. For clarity, this descriptor is replaced with its intensity components in the following expressions. SAD can be defined as:

$$\delta_{SAD}(p, d) = \sum_{q \in N(p)} |I^l(q) - I^r(q - d)| \tag{2.25}$$

- Sum of Squared Difference (SSD) works over intensity neighbourhood descriptor $\xi_W(p)$ as SAD. SSD can be defined as:

$$\delta_{SSD}(p,d) = \sum_{q \in N(p)} (I^l(q) - I^r(q-d))^2 \tag{2.26}$$

- Normalized Cross Correlation (NCC) works over intensity neighbourhood descriptor $\xi_W(p)$ as SAD. NCC compensates for gain changes and is optimal for dealing with Gaussian noise but tends to blur depth discontinuities. NCC can be written as:

$$\delta_{NCC}(p,d) = \frac{\sum_{q \in N(p)} I^l(q) I^r(q-d)}{\sqrt{\sum_{q \in N(p)} I^l(q)^2 \sum_{q \in N(p)} I^r(q-d)^2}} \tag{2.27}$$

- Zero-mean Normalized Cross Correlation (ZNCC) works over intensity neighbourhood descriptor $\xi_W(p)$ as SAD. ZNCC compensates for both offset and gain changes. ZNCC can be written as:

$$\delta_{ZNCC}(p,d) = \frac{\sum_{q \in N(p)}(I^l(q) - \bar{I}^l(p))(I^r(q-d) - \bar{I}^r(p-d))}{\sqrt{\sum_{q \in N(p)}(I^l(q) - \bar{I}^l(p))^2 \sum_{q \in N(p)}(I^r(q-d)^2 - \bar{I}^r(p-d))}} \tag{2.28}$$

#### 2.2.2.3 Disparity Estimation

The calculation of disparity maps requires the selection of the most optimal match for the image elements across the stereo-images. In order to represent how good or bad a match is, a cost function $c(p, p-d)$ is defined where $p$ is a pixel or feature on the left image and $p-d$ is the corresponding pixel on the right image at disparity $d$. Approaches for disparity estimation can be classified, based on the the type of information they use, as: local, global or semi-global.

- *Local Approaches*:- Local approaches use only information from a neighborhood around the matching image element and perform a cost calculation against the elements in the search area. The cost function $c(p, p-d)$ ranges over discrete set of disparities between

$d_{min}$ and $d_{max}$. In the literature, several approaches have been proposed for estimating disparities from the cost volume $V_c(u, v, d)$ constructed over the set of all pixels in the image and all discreet disparities. The optimal disparity $\hat{d}$ at a pixel $p$ can be selected using a Winner Takes All strategy described as

$$\hat{d} = \underset{d \in [d_{min}, d_{max}]}{\arg\min} \; c(p, p - d) \tag{2.29}$$

Aggregation windows $A(p)$, also known as support regions, allow an increase of the uniqueness in descriptors by assuming the disparity is constant in the local neighborhood window. This aggregated cost $c_A$ can be written as

$$c_A(p, p - d) = \sum_{q \in A(p, p-d)} c(q, q - d) \tag{2.30}$$

then the cost $c$ can be replaced by the aggregated cost $c_A$ in the Eq. 2.29 for estimating the optimal disparity $\hat{d}$. Other techniques, such as Left-Right Consistency checks (Hu and Mordohai, 2012), have been used to filter spurious matches in stereo matching tasks. The discrete depth estimation can also be computed with sub-pixel resolution using data term interpolation schemes such as parabola fit. For a more detailed analysis of local approaches, refer to (Newcombe, 2012).

- *Global Approaches*:- Global approaches for disparity calculation use both local information and global priors to define an energy function $E : S \rightarrow \mathbb{R}$ which maps a set of candidate solutions to an optimal solution. This energy function $E$ represents the cost of assigning a disparity value to each image pixel. Most of the energy functions found in the literature have the form

$$E(d) = E_{data}(d) + E_{prior}(d) \tag{2.31}$$

where $d$ represents the disparity values assigned to the image pixels. $E_{data}(d)$, the data term penalizes solutions which are inconsistent to the data and $E_{prior}(d)$ imposes a prior. In most of the literature, the $E_{prior}$ refers to the local neighborhood consistency of the resulting solution. The discrete solution to this formulation corresponds to a labeling problem where the labels correspond to the disparity value of the matched pixels. This can easily be solved using Markov Random Fields models. Although the energy minimization by MRF models is an NP-hard problem, approaches like Belief Propagation (BP) (Felzenszwalb and Huttenlocher, 2006), Graph Cuts (GC), and Dynamic Programming (DP) (Felzenszwalb and Zabih, 2010) have proven to provide good approximations. The second approach formulates the energy minimization in a continuous domain. This kind of formulation is usually optimized by a Total Variation (TV) framework (Pock et al., 2008). The detailed survey on this work can be found in (Newcombe, 2012).

### 2.3 3D Surface Representations

Surface is a two-dimensional manifold in $\mathbb{R}^3$. A manifold can be thought as a smooth geometric figure of a certain dimension. In the literature, several approaches have been discuss for 3D surface representation. All of these approaches can be categorized into three classes namely, *explicit surfaces*, *parametric surfaces* and *implicit surfaces*.

*Explicit surfaces* are defined as a graph over $xy$ - plane i.e. each point on the surface is expressed as $\mathbf{x} = [x, y, f(x, y)]$. In the context of computer vision, the depth maps can be considered as sample data sets from an explicit surface. The shapes that can be easily expressed by this representation are limited because of discontinuity in the graph. For example, overhanging ledges cannot be expressed by this representation.

*Parametric surfaces* are defined by a function $f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$ which maps the parameter space $\Omega$ into 3D space. As surface self-intersections are possible, parametric surfaces like triangle meshes or NURBS surfaces do not generally need to be manifolds. For a surface with complex topology, a consistent and continuous surface and without self-intersections is difficult to obtain by this representation. Although, this representation is efficient to implement and is a common choice to store and represent 3D models in computer graphics.

An *Implicit surface* S is defined as the isocontour of a scalar function $f : \mathbb{R}^3 \to \mathbb{R}$

$$\mathbf{S} = \{\mathbf{x} \in \mathbb{R}^3 | f(\mathbf{x}) = \rho\} \tag{2.32}$$

The set of points S is also called *isosurface* (Bloomenthal et al., 1997). The scalar function $f$ actually defines infinite number of surfaces, one for each *isovalue*. In the context of 3D reconstruction, we always use isocontour $\rho = 0$. For the isosurface to be well defined, it is sufficient that the function $f$ does not have any critical points, i.e., the gradient $\nabla f$ must be defined everywhere and must not be zero. The isosurface partitions space into two sets, the interior and the exterior of the surface. We follow the convention that the interior is the area where the function is negative, the exterior where the function is positive. Thus, the isosurface does not contain any self-intersections. Unlike explicit and parametric surfaces, isosurfaces are always closed, and the notion of interior and exterior naturally establishes a consistent orientation.

In the rest of the section, we discuss the most commonly used *implicit surface* known as Signed Distance Function (SDF). Then we discuss several methods that use *implicit surface* in the context of 3D reconstruction.

### 2.3.1 Signed Distance Function

A special class of scalar functions $f$ to represent the implicit surface has proven to be advantageous for many applications. This function's gradient is enforced to always have a length of one; this can be described by *Eikonal equation*

$$||\nabla f|| = 1 \tag{2.33}$$

Together with the boundary condition of the zero-set $f|_S = 0$, the Eikonal equation 2.33 uniquely defines the function $f$ in $\mathbb{R}^3$. Hence, the two are equivalent definitions. At any point in space, $f$ is the Eucledian distance to the closest point on S, with a negative sign on the inside and a positive sign on the outside of the surface. We visualise this $f$ on the Stanford Bunny 3D model in the Fig.

,

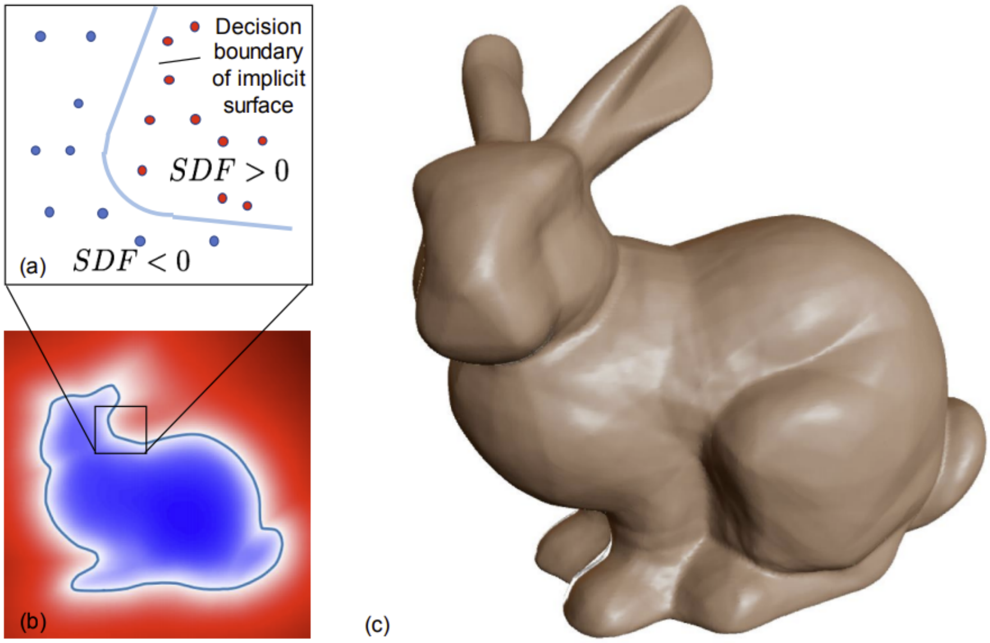


Figure 2.6: Visualization of Signed Distance Function $f$ on the Stanford Bunny 3D model. (a) depiction of the underlying implicit surface with $SDF = 0$ on the surface contour, $SDF < 0$ inside and $SDF > 0$ outside the surface boundary, (b) 2D cross-section of the signed distance field, (c) rendered 3D surface recovered from $SDF = 0$.

### 2.3.2 Surface Representation Methods

To perform tasks such as geometric modeling or just rendering of 3D surfaces on displays, a concrete representation of the implicit surface is necessary. In the literature, a wide range of approaches has been presented. Usually, a trade-off has to be made amongst accuracy, generality, and efficiency, both in terms of memory and computational complexity. Multiple analytic functions can be combined by linear combination or blending to represent more complex functions and surfaces. Several of these basis function approaches have previously been proposed for a wide range of applications. However, the placement of basis functions and choice of appropriate blending weights can be a time-consuming process, regardless of whether they are performed manually or automatically. In digital signal processing, sampling is the most common method for capturing virtually any input data. Both regular and adaptive sampling are common means to represent the scalar function. A reconstruction filter is required to retrieve a continuous definition from the discrete samples. In the following sub-sections, these possible choices of representation will be discussed in greater detail.

#### 2.3.2.1 Radial Basis Function (RBF)

The superposition of translated basis functions is common practice for interpolation and approximation of irregularly sampled data. The set of functions, known as Radial Basis Functions (RBF) are designed to locally represent the shape of the surface and can be moved around freely (Carr et al., 2001). The basis functions in the RBF approach are centered at the sample positions $x$ of the input data. The implicit function $f$ can be defined as

$$f(\mathbf{x}) = \sum_i w_i \phi(||\mathbf{x} - \mathbf{x_i}||) \tag{2.34}$$

The basis function $\phi$ itself is a fixed function of the Euclidean distance to the center point. Common choices for the basis include thin-plate ($r^2 logr$), biharmonic ($r$), and triharmonic ($r^3$) splines, which minimize certain bending energy norms resulting in smooth interpolation. A linear system of equations needs to be solved to determine the appropriate linear combination of basis functions to interpolate the input. More details on the approach can be found in (Buhmann, 2003).

#### 2.3.2.2 Multi-level Partition of Unity (MPU)

Multi-level partition of unity implicits is an alternative representation for reconstructing surfaces from a large set of points (Ohtake et al., 2005). Instead of radial basis functions, three different types of local surface approximations are used: a quadratic polynomial in three dimensions or over a two-dimensional parameter plane and a piece-wise quadratic function to capture sharp features like edges and corners. Those basis functions are fitted locally by analyzing the surface samples, and these local approximations are blended to construct a global definition. Space is adaptively subdivided by an octree according to the local surface detail, with one basis function per octree cell. The blending weights of the partition of unity is defined by a normalized Gaussian around the center of the octree cell.

#### 2.3.2.3 Grid Sampling

A straightforward method to represent arbitrary implicit surfaces is to sample the scalar function on a regular grid. Although many sample values are required to capture fine detail, they can be stored without any overhead in a continuous array that provides fast direct access and can easily be implemented on the GPU hardware. The idea of such volumetric integration of Signed Distance Functions was introduced in (Curless and Levoy, 1996). In this approach, the SDF field can be truncated (TSDF) and estimated only over the region of uncertainty around the input samples. This technique enables sparse allocation of the grid and reduces the memory overhead by explicitly ignoring the free space.

The SDF approximation $F(x)$ at each cell $x$ of the grid can be represented by weighted average of individual SDFs $f_1(x)$, $f_2(x)$, ... $f_n(x)$ and weights $w_1(x)$, $w_2(x)$, ... $w_n(x)$ and can be written as

$$F(x) = \frac{\sum w_i(x) f_i(x)}{\sum w_i(x)} \tag{2.35a}$$

$$W(x) = \sum w_i(x) \tag{2.35b}$$

where, $f_i(x)$ and $w_i(x)$ are the signed distance and weight functions from the $i$th range image. This can be expressed as an incremental calculation as

$$F_i(x) = \frac{W_{i-1}(x) F_{i-1}(x) + w_i(x) d_i(x)}{W_{i-1}(x) + w_i} \tag{2.36a}$$

$$W_i(x) = W_{i-1}(x) + w_i \tag{2.36b}$$

where $F_i(x)$ and $W_i(x)$ are the cumulative signed distance and weight functions after integrating the $i$th range image. KinectFusion (Newcombe et al., 2011a) further extended this approach for real-time fusion of series of unregistered range images. Iterative Closest Point (ICP) was used to register each range map into a consistent volumetric model.

# CHAPTER 3: LEARNING BASED DEPTH ESTIMATION FROM STEREO

## 3.1 Introduction

Depth from stereo vision has been heavily studied in computer vision field for the last few decades. Depth estimation has various applications in autonomous driving, dense reconstruction and 3D objects and human tracking. Virtual Reality and Augmented Reality systems require depth estimations to build dense spatial maps of the environment for interaction and scene understanding. For proper rendering and interaction between virtual and real objects in an augmented 3D world, the depth is expected to be both dense and correct around object boundaries. Depth sensors such as structured light and time of flight sensors are often used to build such spatial maps of indoor environments. These sensors often use illumination sources whose cost exceeds the budget of an envisioned AR system. Since these sensors use infrared vision, they have troubles working in bright sun light environment or in presence of other infrared sources. Although, Time of Flight sensors work outdoors in presence of sunlight but they are not as accurate as structured light sensors and they also suffer from multi-path interference in presence of reflective materials in the environment.

On the other hand, the depth from stereo vision systems have a strong advantage of working in both indoors and in sunlight environments. Since these systems use passive image data, they do not interfere with each other or with the environment materials. Moreover, the resolution of passive stereo systems is typically greater than the sparse patterns used in structured light depth sensors, so these methods have capabilities to produce depth with accurate object boundaries and corners. Due to recent advancements in camera and mobile technology the image sensors have dramatically reduced in size and have significantly improved in resolution and image quality. All these qualities makes passive stereo system a better fit for being a depth estimator for a AR or VR system. However, stereo systems have their own disadvantages, such as ambiguous predictions in texture-less or repeating/confusing textured surfaces. In order to deal with these homogeneous regions traditional methods make use of handcrafted functions and optimize the parameters globally on the entire image. Recent methods use machine learning to derive the functions and it's parameters from the data that is used in training. As these functions tend to be highly non-linear, they tend to yield reasonable approximations even on the homogeneous and reflective surfaces.

### 3.2 Background

Depth from stereo has been widely explored in the literature, we refer interested readers to surveys and methods described in (Scharstein and Szeliski, 2002). Broadly speaking stereo matching can be categorized into computation of cost metrics, cost aggregation, global or semi-global optimization (Hirschmuller, 2008) and refinement or filtering processes. Traditionally global cost filtering approaches used discrete labeling methods such as Graph Cuts (Kolmogorov and Zabih, 2001) or used belief propagation techniques described in (Klaus et al., 2006) and (Bleyer et al., 2011). Total Variation denoising (Rudin et al., 1992) has been used in cost filtering by methods described in (Zach et al., 2007), (Newcombe et al., 2011b) and (Newcombe, 2012).

The state-of-the-art in disparity estimation techniques use CNNs. MC-CNN (Zbontar et al., 2016) introduced a Siamese network to compare two image patches. The scores on matching was used along with the semi-global matching process (Hirschmuller, 2008) to predict consistent disparity estimation. DispNet (Mayer et al., 2016) demonstrates an end-to-end disparity estimation neural network with a correlation layer (dot product of features) for stereo volume construction. (Liang et al., 2018) improved DispNet by introducing novel iterative filtering process. GC-Net (Kendall et al., 2017) introduces a method to filter 4D cost using a 3D cost filtering approach and the soft argmax process to regress depth. PSMNet (Chang and Chen, 2018) improved GC-Net by enriching features with better global context using pyramid spatial pooling process. They also show effective use of stacked residual networks in cost filtering process.

(Xie et al., 2018) introduce vortex pooling which is an improvement of the atrous spatial pooling approach used in Deep lab (Chen et al., 2018). Atrous pooling uses convolutions with various dilation steps to increase receptive fields of a CNN filter. The vortex pooling technique uses average pooling in grids of varying dimensions before dilated convolutions to utilize information from the pixels which were not used in bigger dilation steps. The size of average pool grids grows with the increase in dilation size.

## 3.3 StereoDRNet: Dilated Residual Stereo Net

In this chapter we introduce, StereoDRNet (Chabra et al., 2019), a convolution neural network (CNN) to estimate depth from a stereo pair followed by volumetric fusion of the predicted depth maps to produce a 3D reconstruction of a scene. Our proposed depth refinement architecture, predicts view-consistent disparity and occlusion maps that helps the fusion system to produce geometrically consistent reconstructions. We utilize 3D dilated convolutions in our proposed cost filtering network that yields better filtering while almost halving the computational cost in comparison to state-of-the-art cost filtering architectures. For feature extraction we use the Vortex Pooling architecture (Xie et al., 2018). The proposed method achieves state-of-the-art results in KITTI 2012, KITTI 2015 and ETH 3D stereo benchmarks. Finally, we demonstrate that our system is able to produce high fidelity 3D scene reconstructions that outperforms the state-of-the-art stereo system.

### 3.3.1 Key Contributions

• **Novel Disparity Refinement Network**: The main motivation of our work is to predict geometrically consistent disparity maps for stereo input that can be directly used by TSDF-based fusion system like KinectFusion (Newcombe et al., 2011a) for simultaneous tracking and mapping. Surface normals are an important factor in fusion weight computation in KinectFusion-like systems, and we observed that state-of-the-art stereo systems such as PSMNet produces disparity maps that are not geometrically consistent which negatively affect TSDF fusion. To address this issue, we propose a novel refinement network which takes geometric error $E_g$, photometric error $E_p$ and unrefined disparity as input and produces refined disparity (via residual learning) and the occlusion map. Refinement procedures proposed in CRL (Pang et al., 2017), iResNet (Liang et al., 2018), StereoNet (Khamis et al., 2018) and FlowNet2 (Ilg et al., 2017) only use photometeric error (either in image or feature domain) as part of the input in the refinement networks. To the best of our knowledge we are the first to explore the importance of geometric error and occlusion training for disparity refinement.

• **3D Dilated Convolutions in Cost Filtering**: state-of-the-art stereo systems such as PSMNet (Chang and Chen, 2018) and GC-Net (Kendall et al., 2017) that use 3D cost filtering approach use most of the computational resources in the filtering module of their system. We observe that using 3D dilated convolutions in all three dimensions i.e (width, height, and disparity channels) in a structure shown in Fig. 3.3 gave us better results with less compute (refer to Table.3.4).

• **Other Contributions**: We observe that Vortex Pooling compared to spatial pyramid pooling (used in PSMNet) provides better results (refer to ablation study 3.5). We found the exclusion masks used to filter non-confident regions of ground truth for fine-tuning our model as discussed in Sec 4.2 to be very useful in obtaining sharp edges and fine details in disparity predictions. We achieve 1.3 - 2.1 cm RMSE on 3D reconstructions of three scenes that we prepared using structured light system proposed in (Whelan et al., 2018).

### 3.4 Algorithm

In this section we describe our architecture that predicts disparity for the input stereo pair. Instead of using a generic encoder-decoder CNN we break our algorithm into feature extraction, cost volume filtering and refinement procedures.

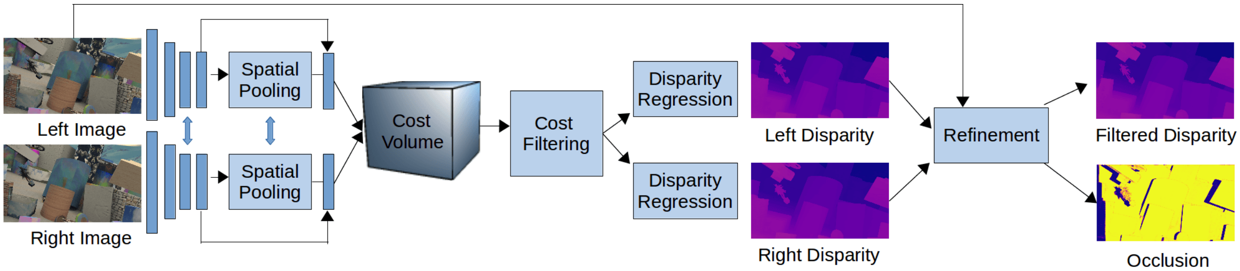


Figure 3.1: StereoDRNet network architecture pipeline.

### 3.4.1 Feature Extraction

The feature extraction starts with a small shared weight Siamese network which takes input as images and encodes the input to a set of features. As these features will be used for stereo matching we want them to have both local and global contextual information. To encode local spatial information in our feature maps we start by downsampling the input by use of convolutions with stride of 2. Instead of having a large $5 \times 5$ convolution we use three $3 \times 3$ filters where first convolution has stride of 2. We bring the resolution to a fourth by having two of such blocks. In order to encode more contextual information we choose Vortex Pooling (Xie et al., 2018) on the learned local feature maps Fig. 3.2. Each of our convolutions are followed by batch normalization and ReLU activation except on the last 3x3 convolution on the spatial pooling output. In order to keep the feature information compact we keep the feature dimension size as 32 throughout the feature extraction process.

### 3.4.2 Cost Volume Filtering

We use the features extracted in the previous step to produce a stereo cost volume. While several approaches in the literature ( (Kendall et al., 2017), (Mayer et al., 2016)) use concatenation or dot products of the stereo features to obtain the cost volume, we found simple arithmetic difference to be just as effective.

While the simple argmin on the cost should in principle lead to the correct local minimum solution, it has been shown several times in literature (Newcombe et al., 2011b), (Hirschmuller,

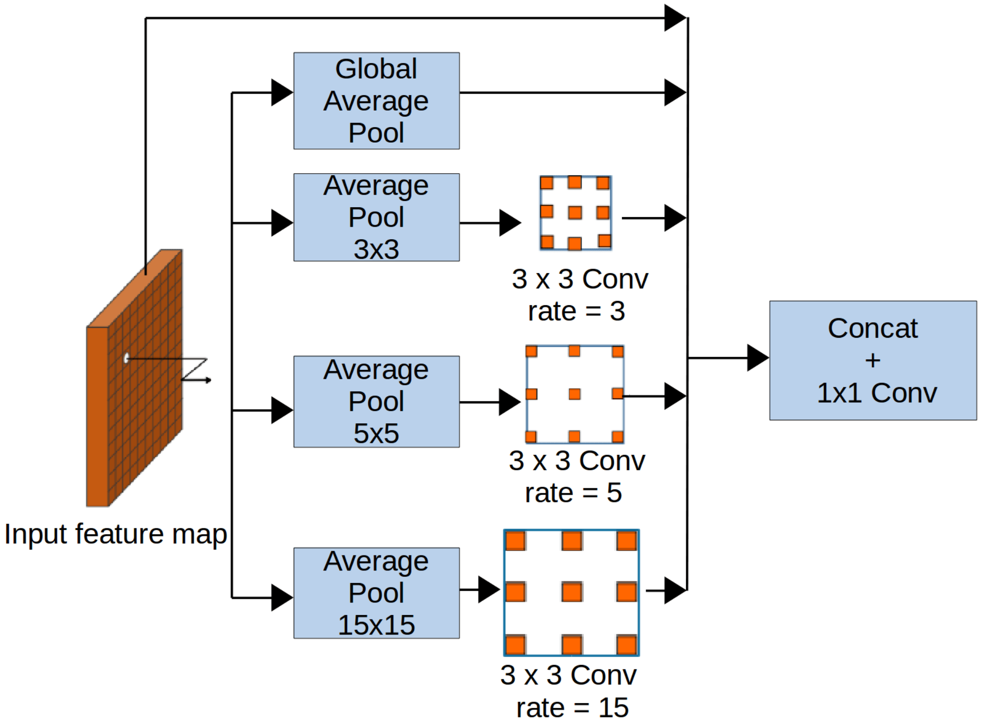


Figure 3.2: StereoDRNet Vortex Pooling architecture derived from (Xie et al., 2018).

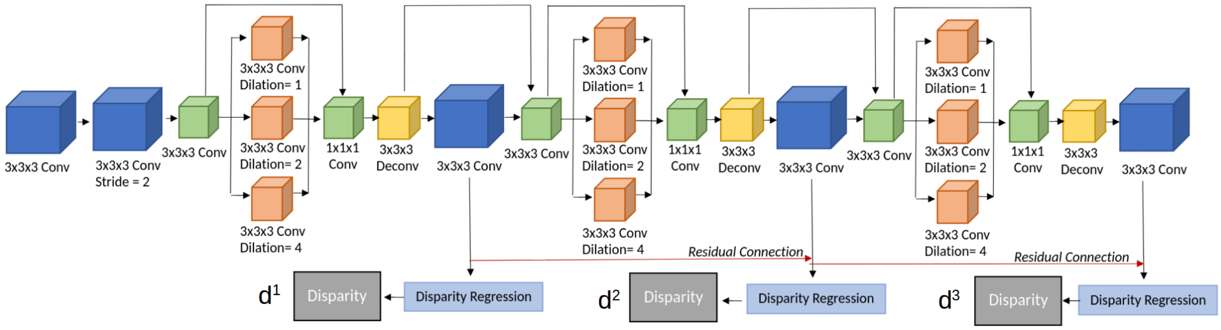


Figure 3.3: Proposed dilated cost filtering approach with residual connections.

2008), (Scharstein and Szeliski, 2002) that it is common for the solution to have several local minima. Surfaces with homogeneous or repeating texture are particularly prone to this problem. By posing the cost filtering as a deep learning process with multiple convolutions and non-linear activations we attempt to resolve these ambiguities and find the correct local minimum.

We start by processing our cost volume with a $3 \times 3 \times 3$ convolution along the width, height and depth dimensions. We then reduce the resolution of the cost by a convolution with stride of 2 followed by convolutions with dilation 1, 2, 4 in parallel. A convolution on the concatenation of the dilated convolution filters is used to combine the information fetched from varying receptive fields.

Residual learning has been shown to be very effective in disparity refinement process so we propose a cascade of such blocks to iteratively improve the quality of our disparity prediction. We depict the entire cost filtering process as Dilated Residual Cost Filtering in Fig. 3.3. In this figure notice how our network is designed to produce $k = 3$ disparity maps labeled as $d^k$.

Our network architecture that supports refinement predicts disparities for both left and right view as separate channels in disparity predictions $d^k$. Note that we construct the cost for both left and right views and concatenate them before filtering; this ensures that the cost filtering method is provided with cost information for both views. Please refer to Table. 3.3 for exact architecture details.

### 3.4.3 Disparity Regression

In order to have a differentiable argmax we use soft argmax as proposed by GC-Net (Kendall et al., 2017). For each pixel $i$ the regressed disparity estimation $d_i$ is defined as a weighted softmax function:

$$d_i = \sum_{d=1}^{N} d \, \frac{e^{-C_i(d)}}{\sum\limits_{d'=1}^{N} e^{-C_i(d')}} \, , \tag{3.1}$$

where $C_i$ is the cost at pixel $i$ and $N$ is the maximum disparity. The loss $L^k$ for each of the proposed disparity maps $d^k$ (as shown in Fig. 3.3) in our dilated residual cost filtering architecture, relies on the Huber loss $\rho$ and is defined as:

$$L^k = \sum_{i}^{M} \rho(d_i^k, \hat{d}_i) \,, \tag{3.2}$$

where $d_i^k$ and $\hat{d}_i$ are the estimated and ground truth disparity at pixel $i$, respectively and $M$ is the total number of pixels. The total data loss $L_d$ is defined as:

$$L_d = \sum_{k=1}^{3} w^k L^k \,, \tag{3.3}$$

where $w^k$ is the weight for each disparity map $d^k$.

### 3.4.4 Disparity Refinement

In order to make the disparity estimation robust to occlusions and view consistency we further optimize the estimate. For brevity we label the third disparity prediction $d^3$ ($k$ = 3) described in Sec. 3.4.2 for left view as $D_l$ and for right view as $D_r$. In our refinement network we warp the right image $I_r$ to left view via the warp $W$ and evaluate the image reconstruction error map $E_p$ for the left image $I_l$ as:

$$E_p = |I_l - W(I_r, D_l)| \,. \tag{3.4}$$

By warping $D_r$ to the left view and using the left disparity $D_l$ we can evaluate the geometric consistency error map $E_g$ as:

$$E_g = |D_l - W(D_r, D_l)| \,. \tag{3.5}$$

While we could just reduce these error terms directly into a loss function, we observed significant improvement by using photo-metric and geometric consistency error maps as input to the refinement network as these error terms are only meaningful for non-occluding pixels (only pixels for which the consistency errors can be reduced).

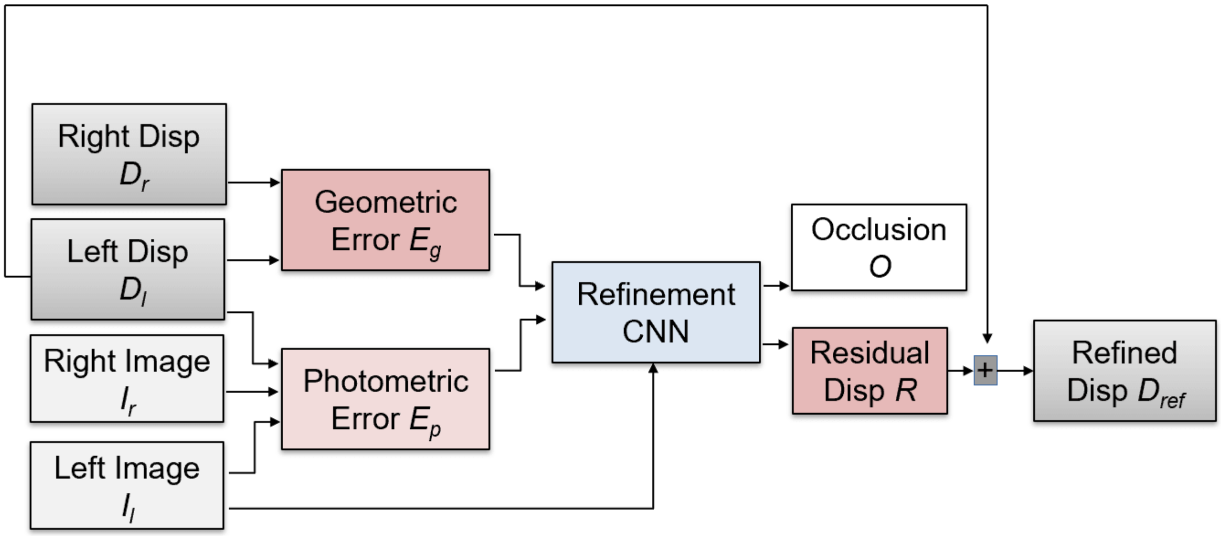


Figure 3.4: StereoDRNet refinement architecture.

Our refinement network takes as input left image $I_l$, left disparity map $D_l$, image reconstruction error map $E_p$ and geometric error map $E_g$. We first filter left image and reconstruction error and left disparity and geometric error map $E_g$ independently by using one layer of convolution followed by batch normalization. Both these results are then concatenated and followed by atrous convolution (Papandreou et al., 2015) to sample from a larger context without increasing the network size. We used dilations with rate 1, 2, 4, 8, 1, and 1 respectively. Finally a single $3 \times 3$ convolution without ReLU or batch normalization is used to output an occlusion map $O$ and a disparity residual map $R$. Our final refined disparity map is labeled as $D_{ref}$. We demonstrate our refinement network in Fig. 3.4 and provide exact architecture details in Table. 3.2.

We compute the cross entropy loss on the occlusion map $O$ as $L_o$

$$L_o = H(O, \hat{O}) \,, \tag{3.6}$$

where $\hat{O}$ is the ground truth occlusion map.

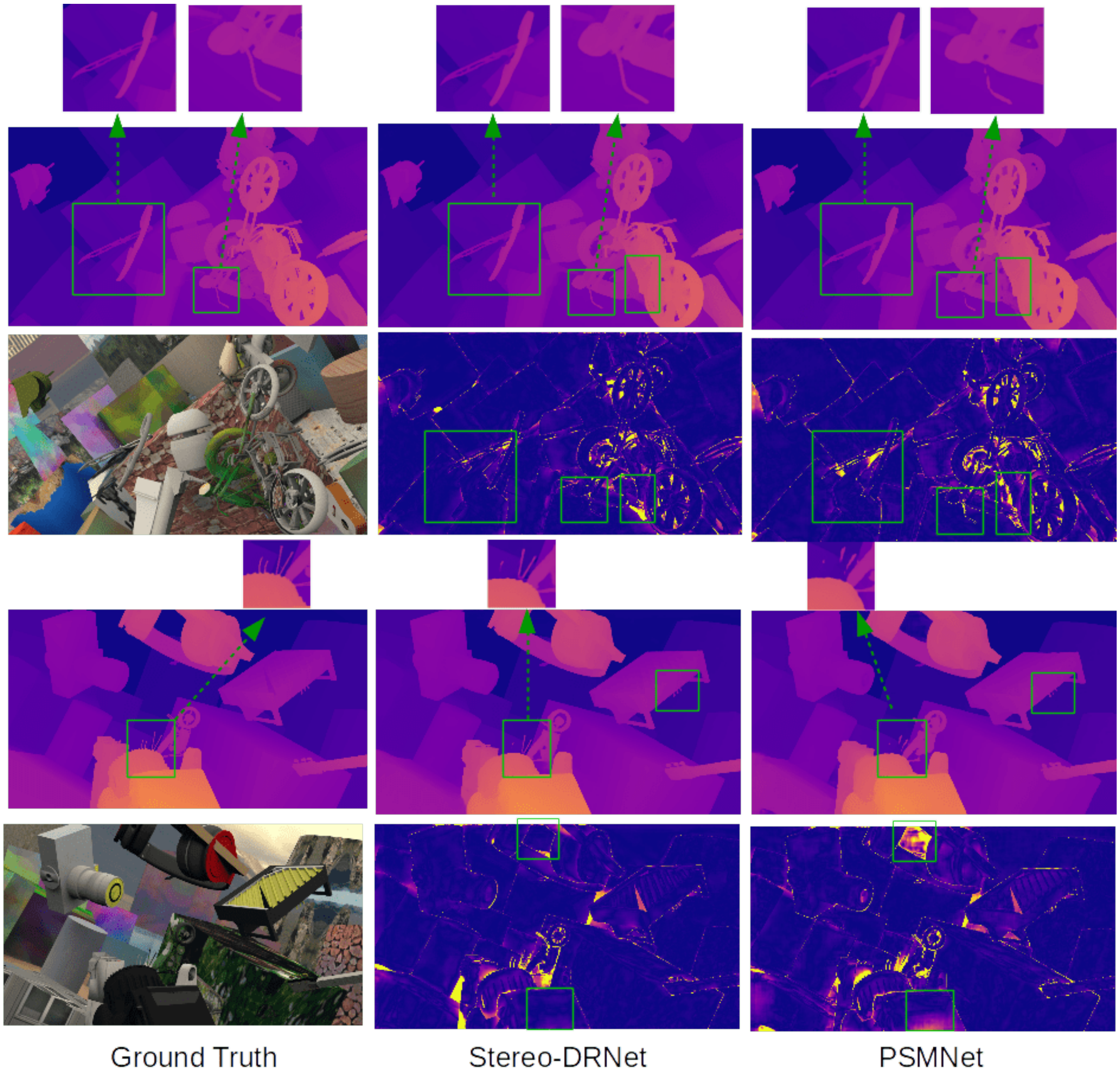


Figure 3.5: Disparity prediction comparison between our network (Stereo-DRNet) and PSMNet (Chang and Chen, 2018) on the SceneFlow dataset. The top row shows disparity and the bottom row shows the EPE map. Note how our network is able to recover thin and small structures and at the same times shows lower error in homogeneous regions.

The refinement loss $L_r$ is defined as

$$L_r = \sum_{i}^{M} \rho(d_i^r, \hat{d}_i)\,, \tag{3.7}$$

where $d_i^r$ is the value for a pixel $i$ in our refined disparity map $D_{ref}$ and $M$ is the total number of pixels.

Our total loss function $L$ is defined as

$$L = L_d + \lambda_1 L_r + \lambda_2 L_o\,, \tag{3.8}$$

where $\lambda_1$ and $\lambda_2$ are scalar weights.

### 3.4.5 Training

We implemented our neural network code in PyTorch. We tried to keep the training of our neural network similar to one described in PSMNet (Chang and Chen, 2018) for ease of comparison. We used Adam optimizer (Kingma and Ba, 2014) with $\beta_1$ = 0.9 and $\beta_2$ = 0.999 and normalized the image data before passing it to the network. In order to optimize the training procedure we cropped the images to 512x256 resolution. For training we used a mini-batch size of 8 on 2 Nvidia Titan-Xp GPUs. We used $w^1$ = 0.2, $w^2$ = 0.4, $w^3$ = 0.6, $\lambda_1$ = 1.2 and $\lambda_2$ = 0.3 weights in our proposed loss functions Eq. 6.4 and Eq. 3.8.

## 3.5 Network Details

We provide the network architecture of StereoDRNet in Table. 3.3. We borrowed ideas on extracting robust local image features from PSMNet (Chang and Chen, 2018). As described in the paper, we use Vortex Pooling (Xie et al., 2018) for extracting global scene context. In our experiments we found dilation rates 3, 5 and 15 and average grids of size $3 \times 3$, $5 \times 5$ and $15 \times 15$

| 3D Dilation in Cost Filtering | | | | SceneFlow |
|---|---|---|---|---|
| rate = 1 | rate = 2 | rate = 4 | rate = 8 | EPE |
| ✓ | | | | 1.13 |
| ✓ | ✓ | | | 1.03 |
| ✓ | ✓ | ✓ | | **0.98** |
| ✓ | ✓ | ✓ | ✓ | 1.01 |

Table 3.1: Ablation study of dilated convolution rates used in the proposed dilated cost filtering scheme. Note that we used StereoDRNet without refinement in this study.

| Index | Layer Description | Output |
|---|---|---|
| 1 | Warp($I_R$,$\mathbf{d_L^3}$) - $I_L$ | H x W x 3 |
| 2 | concat 1, $I_L$ | H x W x 6 |
| 3 | Warp($\mathbf{d_R^3}$, $\mathbf{d_L^3}$) - $\mathbf{d_L^3}$ | H x W x 1 |
| 4 | concat 3, $\mathbf{d_L^3}$ | H x W x 2 |
| 5 | 3x3 conv on 2, 16 features | H x W x 16 |
| 6 | 3x3 conv on 4, 16 features | H x W x 16 |
| 7 | concat 5,6 $I_L$ | H x W x 32 |
| 8-13 | (3x3 conv, residual block) x 6, dil rate 1,2,4,8,1,1 | H x W x 32 |
| 14 | 3x3 conv, 2 features as 14(a) and 14(b) | H x W x 2 |
| 15 | $\mathbf{d^r}$: 14(a) + $\mathbf{d_L^3}$ | H x W |
| 16 | **O**: sigmoid on 14(b) | H x W |

Table 3.2: Refinement network for StereoDRNet. $\mathbf{d^r}$ and **O** represent refined disparity and occlusion probability respectively.

to improve performance more in disparity predictions than the one proposed in the original work for semantic segmentation.

In order to show the effectiveness of the proposed dilated convolutions in cost filtering, we conduct an ablation study in Table. 3.1 on the SceneFlow (Mayer et al., 2016) dataset. We observed that increasing dilation rates improved the quality of predictions. Dilation rates above 4 did not provide any significant gains.

| Index | Layer Description | Output |
|---|---|---|
| 1 | Input Image | H x W x 3 |
| | Local feature extraction | |
| 2 | 3x3 conv, 32 features, stride 2 | H/2 x W/2 x 32 |
| 3-4 | (3x3 conv, 32 features) x 2 | H/2 x W/2 x 32 |

| | | |
|---|---|---|
| 5-7 | (3x3 conv, 32 features, res block) x 3 | H/2 x W/2 x 32 |
| 8 | 3x3 conv, 32 features, stride 2 | H/4 x W/4 x 32 |
| 9-22 | (3x3 conv, 64 features, res block) x 15 | H/4 x W/4 x 64 |
| 23-28 | (3x3 conv, 128 features, res block) x 6 | H/4 x W/4 x 128 |
| Spatial Pooling | | |
| 29 | Global Avg Pool on 28, bi-linear interp | H/4 x W/4 x 128 |
| 30 | Avg Pool 3x3 on 28, conv 3x3, dil rate 3 | H/4 x W/4 x 128 |
| 31 | Avg Pool 5x5 on 28, conv 3x3, dil rate 5 | H/4 x W/4 x 128 |
| 32 | Avg Pool 15x15 on 28, conv 3x3, dil rate 15 | H/4 x W/4 x 128 |
| 33 | Concat 22, 28, 29, 30, 31 and 32 | H/4 x W/4 x 704 |
| 34 | 3x3 conv, 128 features | H/4 x W/4 x 128 |
| 35 | 1 x 1 conv, 32 features without BN and ReLU | H/4 x W/4 x 32 |
| Cost Volume | | |
| 36 | Subtract left 35 from right 35 with D/4 shifts,vice versa | D/4 x H/4 x W/4 x 64 |
| Cost Filtering | | |
| 37-38 | (3x3x3 conv, 32 features) x 2 | D/4 x H/4 x W/4 x 32 |
| 39 | 3x3x3 conv, 32 features, stride 2 | D/8 x H/8 x W/8 x 32 |
| 40 | 3x3x3 conv, 32 features | D/8 x H/8 x W/8 x 32 |
| 41 | 3x3x3 conv on 39, 32 features | D/8 x H/8 x W/8 x 32 |
| 42 | 3x3x3 conv on 39, 32 features, dil rate 2 | D/8 x H/8 x W/8 x 32 |
| 43 | 3x3x3 conv on 39, 32 features, dil rate 4 | D/8 x H/8 x W/8 x 32 |
| 44 | 3x3x3 conv on concat(41,42,43), 32 features | D/8 x H/8 x W/8 x 32 |
| 45 | 3x3x3 deconv, 32 features, stride 2 | D/4 x H/4 x W/4 x 32 |

| | | |
|---|---|---|
| 46 | **Pred1**: 3x3x3 conv on 45 + 38 | D/4 x H/4 x W/4 x 2 |
| 47 | 3x3x3 conv on 45, 32 features, stride 2 | D/8 x H/8 x W/8 x 32 |
| 48 | 3x3x3 conv + 40, 32 features | D/8 x H/8 x W/8 x 32 |
| 49 | 3x3x3 conv on 48, 32 features | D/8 x H/8 x W/8 x 32 |
| 50 | 3x3x3 conv on 48, 32 features, dil rate 2 | D/8 x H/8 x W/8 x 32 |
| 51 | 3x3x3 conv on 48, 32 features, dil rate 4 | D/8 x H/8 x W/8 x 32 |
| 52 | 3x3x3 conv on concat(49,50,51), 32 features | D/8 x H/8 x W/8 x 32 |
| 53 | 3x3x3 deconv, 32 features, stride 2 | D/4 x H/4 x W/4 x 32 |
| 54 | **Pred2**: 3x3x3 conv on 53 + 38 | D/4 x H/4 x W/4 x 2 |
| 55 | 3x3x3 conv on 53, 32 features, stride 2 | D/8 x H/8 x W/8 x 32 |
| 56 | 3x3x3 conv + 48, 32 features | D/8 x H/8 x W/8 x 32 |
| 57 | 3x3x3 conv on 56, 32 features | D/8 x H/8 x W/8 x 32 |
| 58 | 3x3x3 conv on 56, 32 features, dil rate 2 | D/8 x H/8 x W/8 x 32 |
| 59 | 3x3x3 conv on 56, 32 features, dil rate 4 | D/8 x H/8 x W/8 x 32 |
| 60 | 3x3x3 conv on concat(57,58,59), 32 features | D/8 x H/8 x W/8 x 32 |
| 61 | 3x3x3 deconv, 32 features, stride 2 | D/4 x H/4 x W/4 x 32 |
| 62 | **Pred3**: 3x3x3 conv on 61 + 38 | D/4 x H/4 x W/4 x 2 |
| Disparity Regression | | |
| 63 | Bi-linear interp of **Pred1**, **Pred2**, **Pred3** | D x H x W x 2 |
| 64 | SoftArg Max of 63 to get $\mathbf{d^1}$, $\mathbf{d^2}$, $\mathbf{d^3}$ | H x W x 2 |

Table 3.3: Full StereoDRNet architecture. Note that when used without refinement, StereoDRNet just outputs $\mathbf{d^1}$, $\mathbf{d^2}$ and $\mathbf{d^3}$ for the left view.

The proposed refinement network described in Table. 3.2 is inspired by the refinement procedures proposed in CRL (Pang et al., 2017), iResNet (Liang et al., 2018), StereoNet (Khamis et al., 2018), and ActiveStereoNet (Zhang et al., 2018). We adopted the basic architecture for refinement as described in StereoNet (Khamis et al., 2018) with dilated residual blocks (Yu et al., 2017) to increase the receptive field of filtering without compromising resolution. This technique was also adopted in recent work on optical flow prediction Pwc-net (Sun et al., 2018). We experienced additional gains when using the photometric error $E_p$ and geometric error maps $E_g$ as inputs and co-training of occlusion maps. Such enhancements in the refinement procedure has never been proposed to the best of our knowledge.

## 3.6 Experiments

We tested our architecture on rectified stereo datasets such as SceneFlow, KITTI 2012, KITTI 2015 and ETH3D. We also demonstrate the utility of our system in building 3D reconstruction of indoor scenes.

### 3.6.1 SceneFlow Dataset

SceneFlow (Mayer et al., 2016) is a synthetic dataset with over $30,000$ stereo pairs for training and around $4000$ stereo pairs for evaluation. We use both left and right ground truth disparities for training our network. We compute the ground truth occlusion map by defining as occluded any pixel with disparities inconsistency larger than 1 px. This dataset is challenging due to presence of occlusions, thin structures and large disparities.

In Fig. 3.5 we visually compare our results with PSMNet (Chang and Chen, 2018). Our system infers better structural details in the disparity image and also produces consistent depth maps with significantly less errors in homogeneous regions. We further visualize the effect of our refinement network in Fig. 4.9.

Table 3.4 shows a quantitative analysis of our architecture with and without refinement network. Stereo-DRNet achieves significantly lower end point error while reducing computation time. Our proposed cost filtering approach achieves better accuracy with significantly less compute, demonstrating the effectiveness of the proposed dilated residual cost filtering approach.

| Method | EPE | Total FLOPS | 3D-Conv FLOPS | FPS |
|---|---|---|---|---|
| CRL (Pang et al., 2017) | 1.32 | - | - | 2.1 |
| GC-Net (Kendall et al., 2017) | 2.51 | 8789 GMac | 8749 GMac | 1.1 |
| PSMNet (Chang and Chen, 2018) | 1.09 | 2594 GMac | 2362 GMac | 2.3 |
| Ours | 0.98 | **1410 GMac** | **1119 GMac** | **4.3** |
| Ours-Refined | **0.86** | 1711 GMacs | 1356 GMacs | 3.6 |

Table 3.4: Quantitative comparison of the proposed Stereo-DRNet with the state-of-the-art methods on the SceneFlow dataset. EPE represent the mean end point error in disparity. FPS and FLOPS (needed by the convolution layers) are measured on full $960 \times 540$ resolution stereo pairs. Notice even our unrefined disparity architecture outperforms the state-of-the-art method PSMNet (Chang and Chen, 2018) while requiring significantly less computation.

| Network Architecture | | | | | | | SceneFlow | KITTI-2015 |
|---|---|---|---|---|---|---|---|---|
| Pooling | Cost Filtering | | | Refinement | | | EPE | Val Error(%) |
| | $d^1$ | $d^2$ | $d^3$ | $E_p$ | $E_g$ | $L_o$ | | |
| Pyramid | ✓ | | | | | | 1.17 | 2.28 |
| Vortex | ✓ | | | | | | 1.13 | 2.14 |
| Vortex | ✓ | ✓ | | | | | 0.99 | 1.88 |
| Vortex | ✓ | ✓ | ✓ | | | | 0.98 | **1.74** |
| Pyramid | ✓ | ✓ | ✓ | | | | 1.00 | 1.81 |
| Vortex | ✓ | ✓ | ✓ | ✓ | | | 1.03 | - |
| Vortex | ✓ | ✓ | ✓ | | ✓ | | 0.95 | - |
| Vortex | ✓ | ✓ | ✓ | ✓ | ✓ | | 0.93 | - |
| Vortex | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | **0.86** | - |
| Pyramid | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 0.96 | - |

Table 3.5: Ablation study of network architecture settings on SceneFlow and KITTI-2015 evaluation dataset.

**Ablation study:** In Table 3.5 we show a complete EPE breakdown for different parts of our network on the SceneFlow dataset. Both vortex pooling and refinement procedure add marginal performance gains. Co-training occlusion map with residual disparity drastically improves the mean end point disparity error of the final disparity from 0.93 px to 0.86 px. Passing only the photometric error into the refinement network actually degrades the performance.

### 3.6.2 KITTI Datasets

We evaluated our method on both KITTI 2015 and KITTI 2012 datasets. These data sets contain stereo pairs with semi-dense depth images acquired using a LIDAR sensor that can be

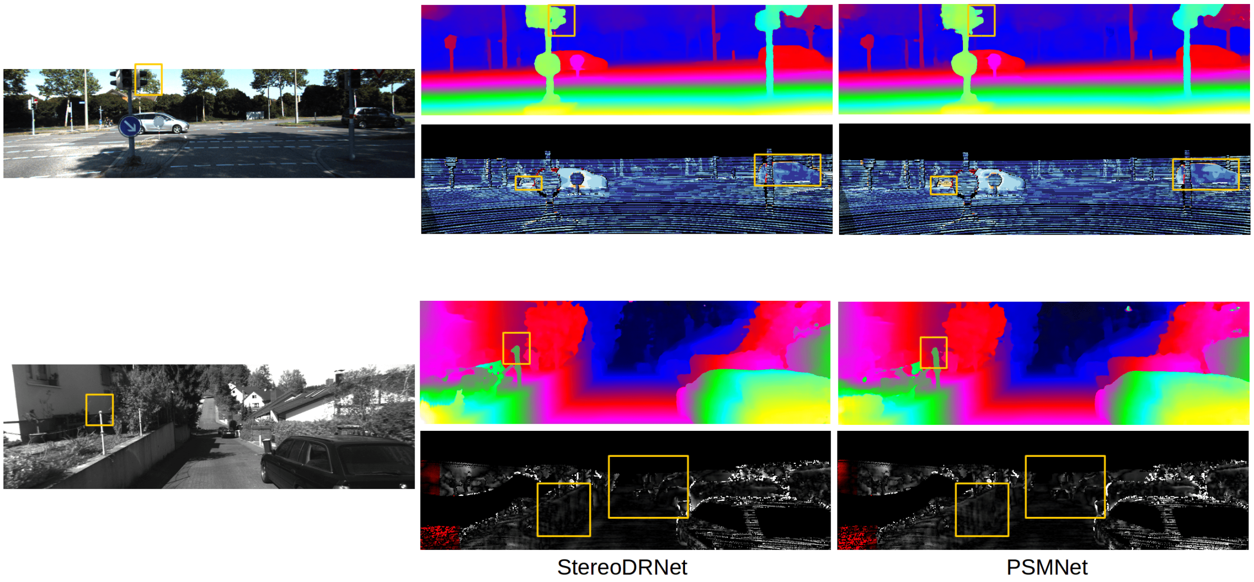


Figure 3.6: This figure shows the disparity estimation results of our StereoDRNet and PSMNet (Chang and Chen, 2018) on the KITTI 2015 and the KITTI 2012 dataset.

used for training. The KITTI 2012 dataset contains 194 training and 193 test stereo image pairs from static outdoor scenes. The KITTI 2015 dataset contains 200 training and 200 test stereo image pairs from both static and dynamic outdoor scenes.

**Training and ablation study:** Since KITTI data sets contain only limited amount of training data, we fine tuned our model on the SceneFlow dataset. In our training we used 80% stereo pairs for training and 20% stereo pairs for evaluation. We demonstrate the ablation study of our proposed method on KITTI 2015 dataset Table 3.5. Note how our proposed dilated residual architecture and the use of Vortex pooling for feature extraction consistently improve the results. We did not achieve significant gains by doing refinement on KITTI datasets as these datasets only contain labeled depth for sparse pixels. Our refinement procedure improves disparity predictions using view consistency checks and sparsity in ground truth data affected the training procedure. We demonstrate that data sets with denser training data enabled the training and fine-tuning of our refinement model.

**Results:** We evaluated our Dilated residual network without filtering on both these datasets and achieved state-of-the-art results on KITTI 2012 Table 3.6 and comparable results with best published method on KITTI 2015 Table 3.7. On KITTI 2015 dataset the three columns “D1-bg”,

| Method | 2px | | 3px | | Avg Error | | Time(s) |
|---|---|---|---|---|---|---|---|
| | Noc | All | Noc | All | Noc | All | |
| GC-NET (Kendall et al., 2017) | 2.71 | 3.46 | 1.77 | 2.30 | 0.6 | 0.7 | 0.90 |
| EdgeStereo (Song et al., 2018) | 2.79 | 3.43 | 1.73 | 2.18 | 0.5 | 0.6 | 0.48 |
| PDSNet (Tulyakov et al., 2018) | 3.82 | 4.65 | 1.92 | 2.53 | 0.9 | 1.0 | 0.50 |
| SegStereo (Yang et al., 2018) | 2.66 | 3.19 | 1.68 | 2.03 | 0.5 | 0.6 | 0.60 |
| PSMNet (Chang and Chen, 2018) | 2.44 | 3.01 | 1.49 | 1.89 | 0.5 | 0.6 | 0.41 |
| Ours | **2.29** | **2.87** | **1.42** | **1.83** | **0.5** | **0.5** | **0.23** |

Table 3.6: Comparison of disparity estimation from StereoDRNet with state-of-the-art published methods on KITTI 2012 dataset.

| Method | All(%) | | | Noc(%) | | | Time(s) |
|---|---|---|---|---|---|---|---|
| | D1-bg | D1-fg | D1-all | D1-bg | D1-fg | D1-all | |
| DN-CSS (Ilg et al., 2018b) | 2.39 | 5.71 | 2.94 | 2.23 | 4.96 | 2.68 | **0.07** |
| GC-NET (Kendall et al., 2017) | 2.21 | 6.16 | 2.87 | 2.02 | 5.58 | 2.61 | 0.90 |
| CRL (Pang et al., 2017) | 2.48 | **3.59** | 2.67 | 2.32 | **3.12** | 2.45 | 0.47 |
| EdgeStereo (Song et al., 2018) | 2.27 | 4.18 | 2.59 | 2.12 | 3.85 | 2.40 | 0.27 |
| PDSNet (Tulyakov et al., 2018) | 2.29 | 4.05 | 2.58 | 2.09 | 3.69 | 2.36 | 0.50 |
| PSMNet (Chang and Chen, 2018) | 1.86 | 4.62 | 2.32 | 1.71 | 4.31 | 2.14 | 0.41 |
| SegStereo (Yang et al., 2018) | 1.88 | 4.07 | **2.25** | 1.76 | 3.70 | 2.08 | 0.60 |
| Ours | **1.72** | 4.95 | 2.26 | **1.57** | 4.58 | **2.06** | 0.23 |

Table 3.7: Comparison of disparity estimation from StereoDRNet with state-of-the-art published methods on KITTI 2015 dataset.

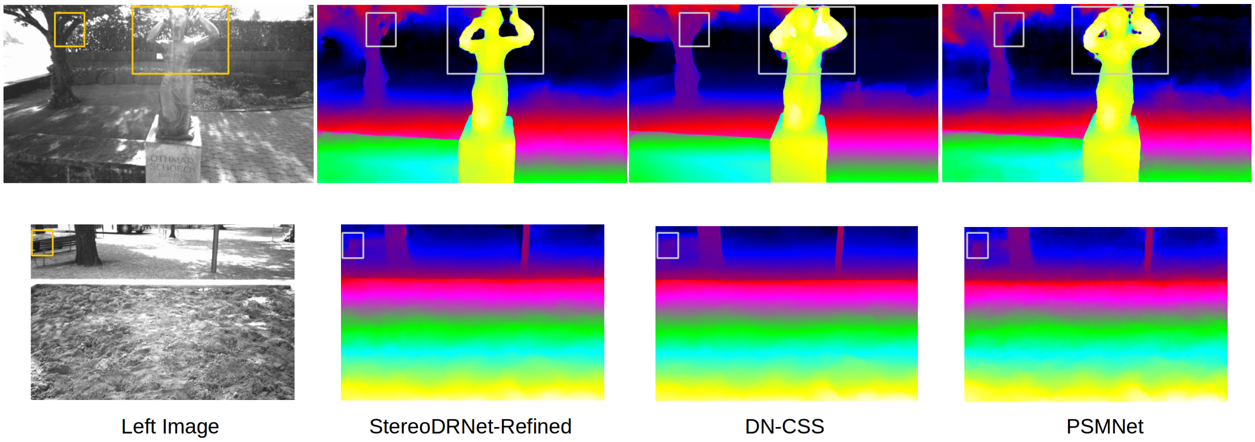
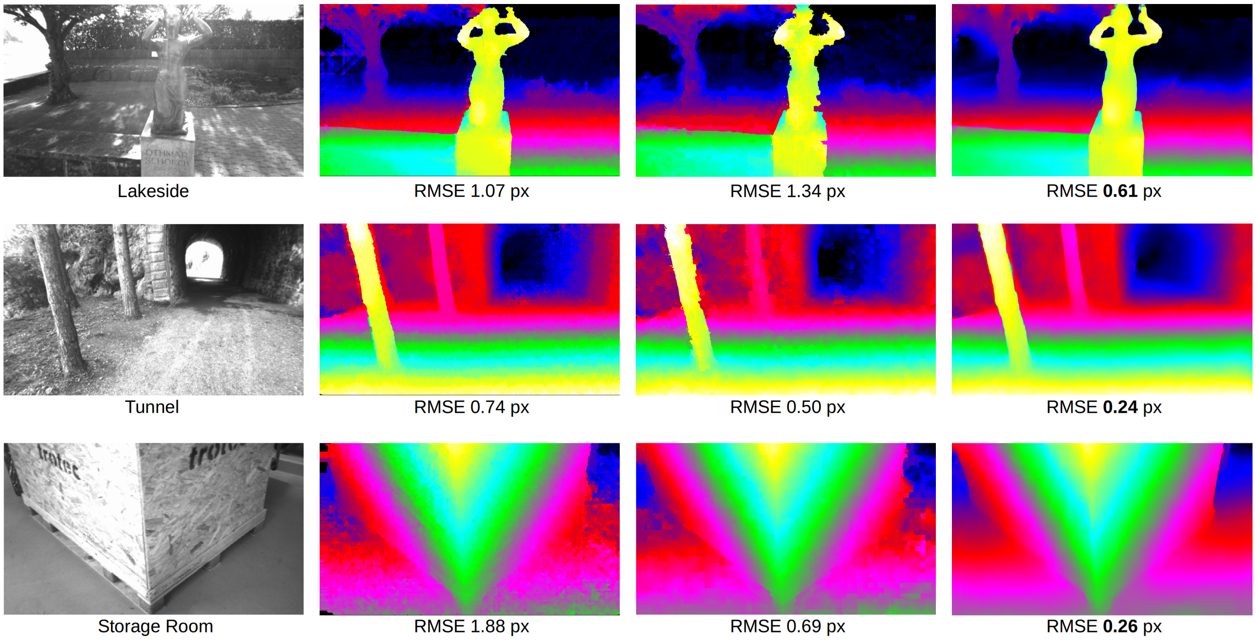


Figure 3.7: This figure shows the disparity estimation results of our refined network, PSMNet (Chang and Chen, 2018) and DN-CSS (Ilg et al., 2018b) on the lakeside and sandbox scenes from the ETH3D (Schöps et al., 2017) two view stereo dataset.

| Method | All | | | | Noc | | | |
|---|---|---|---|---|---|---|---|---|
| | 1px | 2px | 4px | RMSE | 1px | 2px | 4px | RMSE |
| SGM (Hirschmuller, 2008) | 10.77 | 4.67 | 2.03 | 2.11 | 10.08 | 4.07 | 1.54 | 1.89 |
| MeshStereo (Zhang et al., 2015) | 11.94 | 6.09 | 2.79 | 1.29 | 11.52 | 5.78 | 2.61 | 1.21 |
| PSMNet (Chang and Chen, 2018) | 5.41 | 1.31 | 0.54 | 0.75 | 5.02 | 1.09 | 0.41 | 0.66 |
| iResNet (Liang et al., 2018) | 4.04 | 1.20 | 0.34 | 0.59 | 3.68 | 1.00 | 0.25 | 0.51 |
| DN-CSS (Ilg et al., 2018b) | **3.00** | 0.96 | 0.34 | 0.56 | **2.69** | **0.77** | 0.26 | **0.48** |
| Ours | 4.84 | **0.96** | **0.30** | **0.55** | 4.46 | 0.83 | **0.24** | 0.50 |

Table 3.8: Comparison of disparity estimation from StereoDRNet with state-of-the-art published methods on ETH 3D dataset.

“D1-fg” and “D1-all” mean that the pixels in the background, foreground, and all areas, respectively, were considered in the estimation of errors. We perform consistently well in “D1-bg” meaning background areas, we achieve comparable results with state of art method in all pixels and better results in non-occluded regions. On KITTI 2012 dataset ”Noc” means non occluded regions and ”All” mean all regions. Notice, that we perform comparable against SegStereo (Yang et al., 2018) on KITTI 2015 but way better in KITTI 2012 dataset. We provide qualitative evaluation of the KITTI Benchmark results in  Fig.3.6



(a) Left Image (b) SGM (c) MeshStereo (d) StereoDRNet

Figure 3.8: This figure shows the disparity estimation results of our refined network and several other traditional algorithms such as SGM (Hirschmuller, 2008) and MeshStereo (Zhang et al., 2015) on three scenes from the ETH3D (Schöps et al., 2017). Notice, that StereoDRNet produces less error on object boundaries and maintains regularization in the prediction.

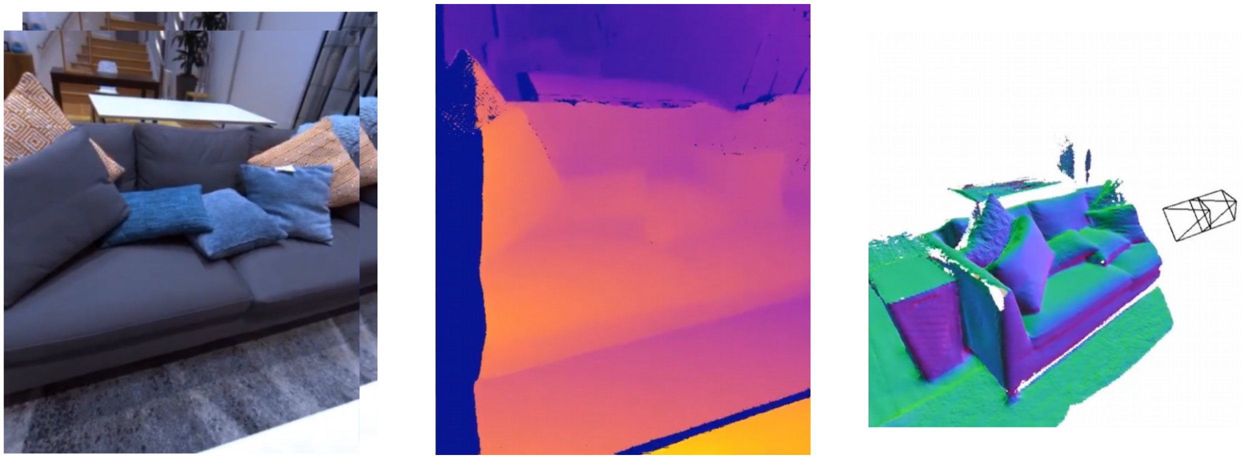
### 3.6.3 ETH3D Dataset

We again used our pre-trained network trained on Sceneflow dataset and fine-tuned it on the training set provided in the dataset. ETH dataset contains challenging scenes of both outside and indoor environment. According to our Table 3.8 we perform best on almost half of the evaluation metrics, our major competitor in this evaluation was DN-CSS (Ilg et al., 2018b). Although, we observe that this method did not perform well on KITTI 2015 data set Table 3.7. Notice, as this data set contained dense training disparity maps of both stereo views we were able to train and evaluate our refinement network on this data set. We provide qualitative evaluation of the ETH3D Benchmark results in Fig.3.7. Furthermore, we provide qualitative and quantitative comparison with other popular traditional stereo matching algorithms such as SGM (Hirschmuller, 2008) and MeshStereo (Zhang et al., 2015) in Fig. 3.8 and Table 3.8

## 3.7 Conclusion

Depth estimation from passive stereo images is a challenging task. Systems from related work suffer in regions with homogeneous texture or surfaces with shadows and specular reflections. Our proposed network architecture, StereoDRNet uses global spatial pooling and dilated residual cost filtering techniques to approximate the underlying geometry even in above mentioned challenging scenarios. Furthermore, our refinement network produces geometrically consistent disparity maps with the help of occlusion and view consistency cues. In the next chapter we will discuss how the proposed depth estimation framework can be used to obtain the 3D reconstruction of challenging indoor scenes.

# CHAPTER 4: 3D RECONSTRUCTION FROM LEARNING BASED STEREO



(a) Input Stereo Images

(b) Predicted Depth

(c) 3D Reconstruction

Figure 4.1: This figure shows the proposed pipeline from input color images to predicted depth from StereoDRNet (Chabra et al., 2019) to 3D Reconstruction prepared using method described in KinectFusion (Newcombe et al., 2011a).

## 4.1 Introduction

In this chapter, we discuss the possibility of using learning based depth from stereo, StereoDRNet (Chabra et al., 2019) for 3D scene reconstruction task. In the literature, several multi-view stereo systems (Schönberger et al., 2016) have been proposed where depth map for an image is generated by matching multiple images from an unstructured set of input images. These depth maps are then fused in a consistent 3D reconstruction model. In contrast, in this chapter we discuss an alternate procedure where depth maps are generated from two images of a stereo camera but the 3D reconstruction is generated by volumeteric TSDF fusion (Curless and Levoy, 1996) of the entire capture sequence. The poses of each depth frame can be obtained from offline, Structured from Motion (Schönberger and Frahm, 2016) systems. Although, if the depth maps predicted are dense, geometrically and temporally consistent then a cheaper algorithm, it-

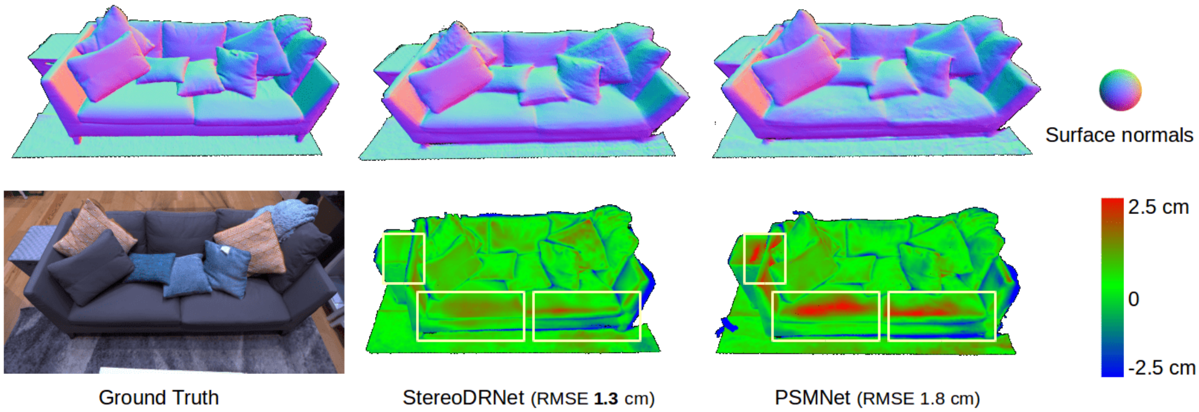


Figure 4.2: StereoDRNet enables estimation of high quality depth maps that opens the door to high quality reconstruction by passive stereo video. In this figure we compare the output from dense reconstruction (Newcombe et al., 2011a) built form depth maps generated by StereoDRNet, PSMNet (Chang and Chen, 2018) and a structured light system (Whelan et al., 2018) (termed Ground Truth). We report and visualize point-to-plane distance RMS error on the reconstructed meshes with respect to the ground truth demonstrating the improvement in reconstruction over the state-of-the-art.

erative closest points (ICP) can be used to approximate camera poses of each depth frame. This procedure has been shown to work in KinectFusion (Newcombe et al., 2011a) where depth was obtained from a structured light depth sensor. In this chapter, we discuss that a learning based depth estimation from a stereo pair, StereoDRNet (Chabra et al., 2019) can be used to build 3D reconstruction for large indoor scene environments in challenging scenarios. As the predicted depth is dense and geometrically consistent so TSDF fusion with ICP based 3D pose estimation can be used. Furthermore, the proposed method is compared against both two-view and multi-view stereo state-of-the-art systems.

## 4.2 Indoor Scene Reconstruction

We use the scanning rig used in recent work (Whelan et al., 2018) for preparing ground truth dataset for supervised learning of depth and added one more RGB camera to the rig to obtain a stereo image pair. We kept the baseline of the stereo pair to be about 10cm. We trained our StereoDRNet network on SceneFlow as described in section 4.1 and then fine tuned the pre-trained network on 250 stereo pairs collected in the indoor area by our scanning rig as shown

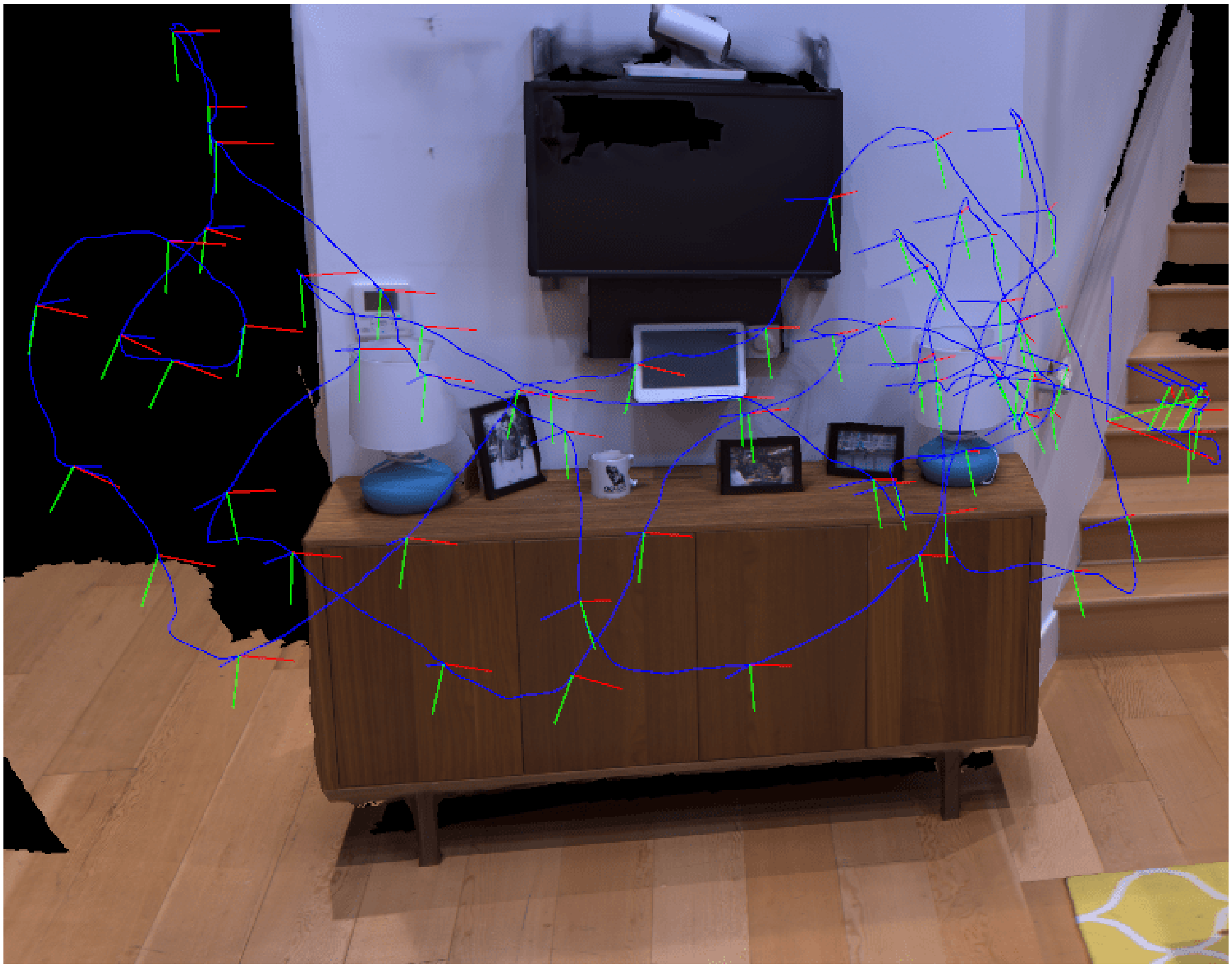


Figure 4.3: This figure shows the training scene used for all our indoor scene reconstruction experiments. We used about 200 real stereo images in real-scene training experiment, these views are shown in this figure by 3D axis along with the camera trajectory visualized by the blue curve. The 3D reconstruction was built using the method described in (Whelan et al., 2018).

in Fig. 4.3. We observed that the network to quickly adapted to our stereo rig with a minimal amount of fine-tuning on SceneFlow (Mayer et al., 2016).

For preparing ground truth depth we found rendered depth from complete scene reconstruction to be a better estimate than the live sensor depth which usually suffers from occlusions and depth uncertainties. Truncated signed distance function (TSDF) was used to fuse live depth maps into a scene as described in (Newcombe et al., 2011a).

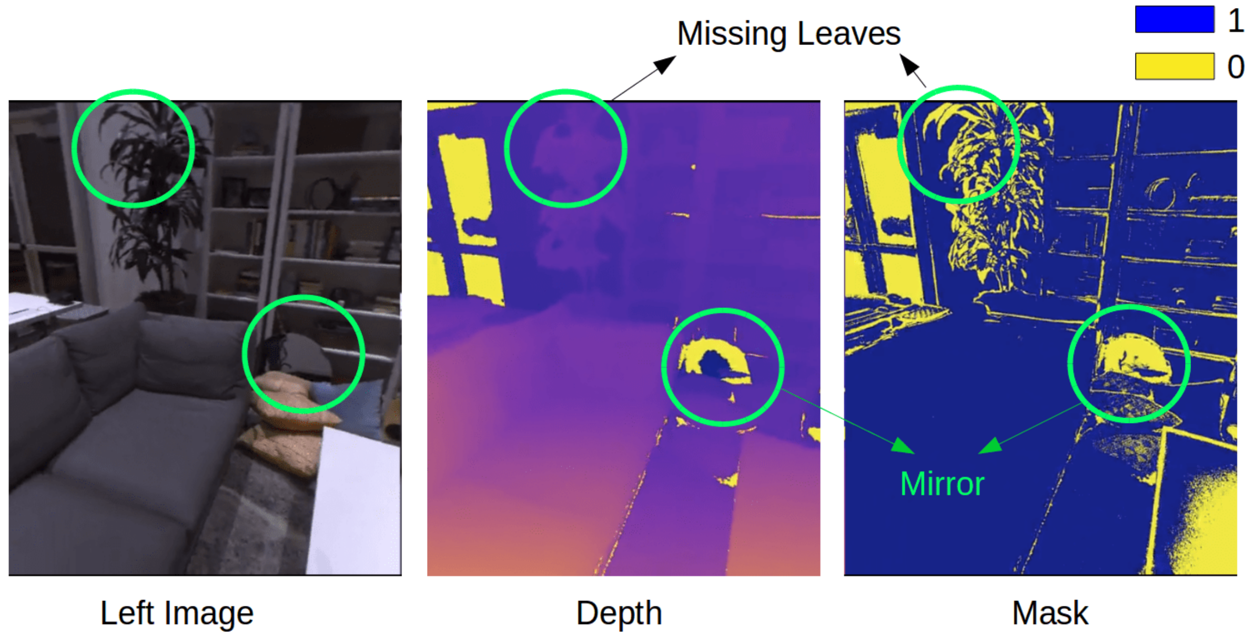


Figure 4.4: We show a training example with the left image, ground truth depth and the exclusion mask. Note that the glass, mirrors and the sharp corners of the table are excluded from training as indicated by the yellow pixels in the occlusion mask. Note, that this example was not part of our actual training set.

The infrared-structure light depth sensors are known to be unresponsive to dark and highly reflective surfaces. Moreover, the quality of TSDF fusion is limited to the resolution of the voxel size. Hence we expect the reconstructions to be overly smooth in some areas such as table corners or sharp edges of plant leaves. In order to avoid contaminating our training data with false depth estimation, we use a simple photometric error threshold to mask out the pixels from training where the textured model projection color disagrees with the real images. We show one such example in Fig. 4.4 where glass, mirrors and the sharp corners of the table are excluded from

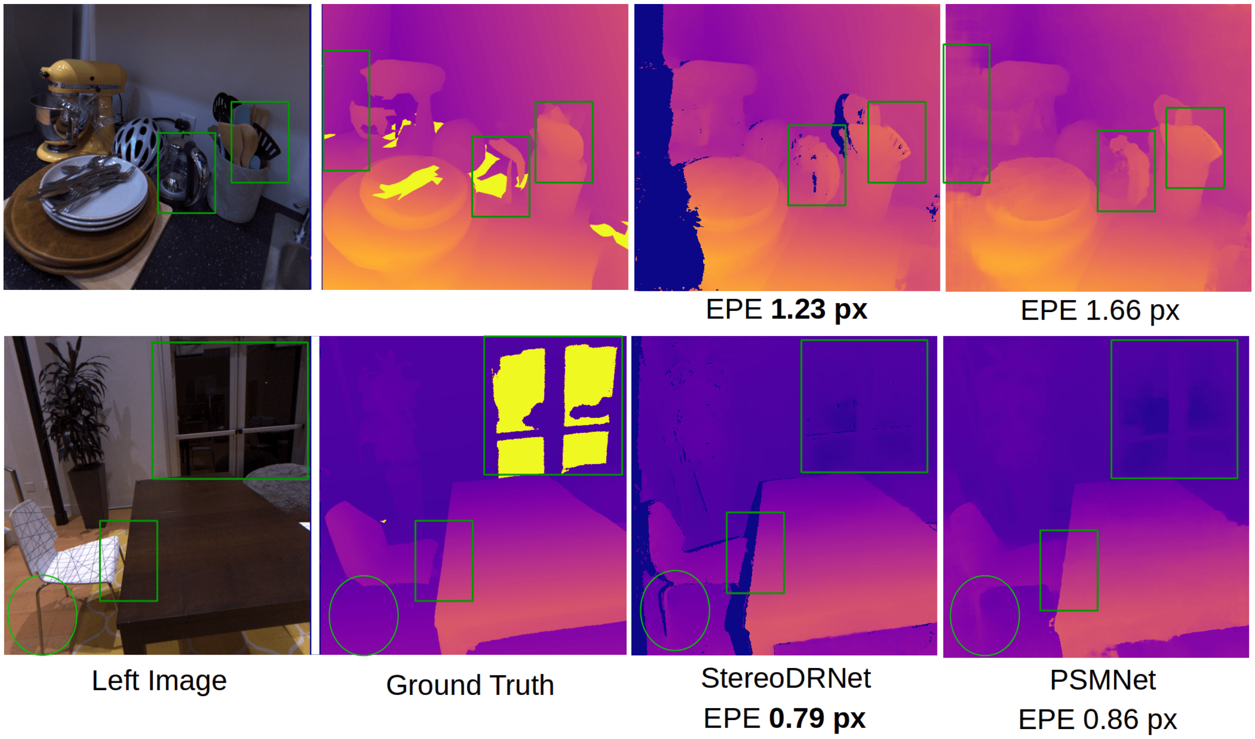


Figure 4.5: This figure demonstrates that our StereoDRNet network produces better predictions on thin reflective legs of the chair and some portions of the glass. We used occlusion mask predicted by our network to clip occluding regions. Yellow region in the ground truth are the regions that belong to our proposed exclusion mask.

training. Although, the system from (Whelan et al., 2018) can obtain ground truth planes of mirrors and glass we avoid depth supervision on them in this work as it is beyond the scope of a stereo matching procedure to obtain depth on reflectors.

We demonstrate visualizations of the depth predictions from the stereo pair in Fig. 4.5. Notice, our prediction is able to recover sharp corners of the table, thin reflective legs of the chair and several thin structures in kitchen dataset as a result of filtering process used in training. It is interesting to see that we recover the top part of the glass correctly but not the bottom part of the glass which suffers from reflections. The stereo matching model simply treats reflectors as windows in presence of reflections.

**Results and evaluations:** We demonstrate visualizations of full 3D reconstruction of a living room in an apartment prepared by TSDF fusion of the predicted depth maps from our system in Fig. 4.6. For evaluation study we prepared three small data sets that we refer as “Sofa and

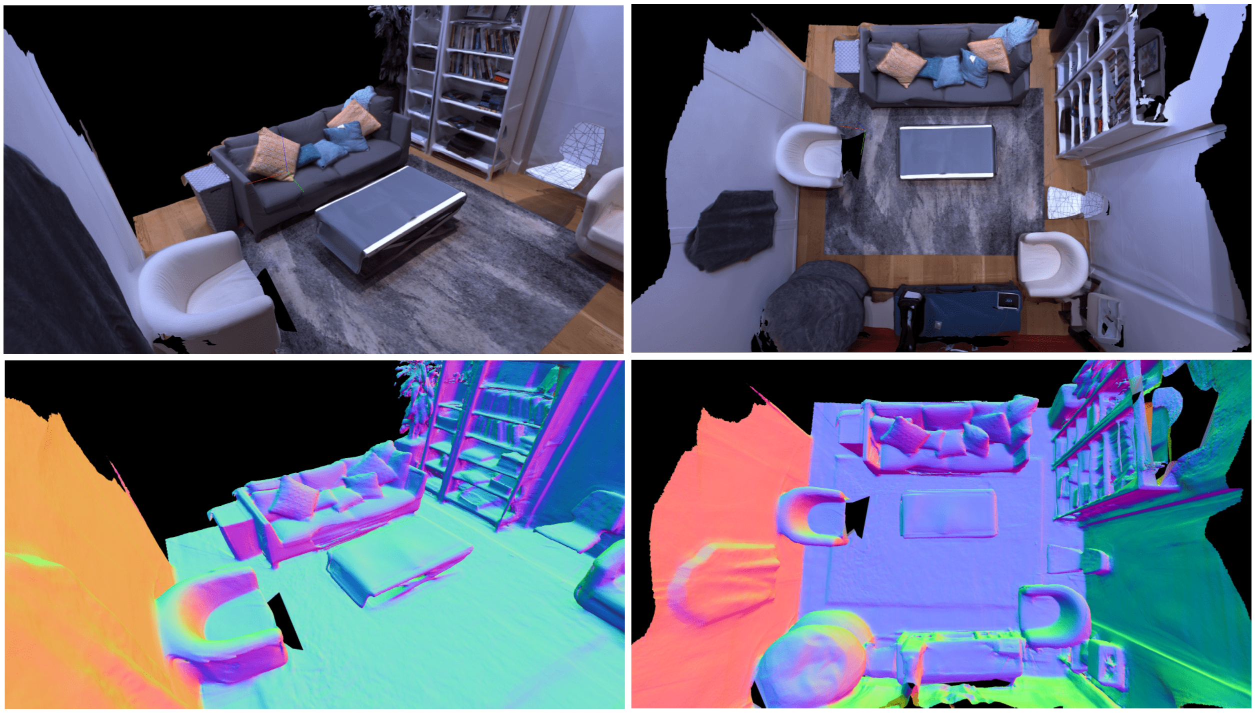


Figure 4.6: This figure demonstrates 3D reconstruction of a living room in an apartment prepared by TSDF fusion of the predicted depth maps from our system. We visualize two views of the textured mesh and surface normals in top and bottom rows respectively.

cushions” demonstrated in Fig. 4.2, “Plants and couch” and “Kitchen and bike” demonstrated in Fig. 4.7. We report point-to-plane root mean squared error (RMSE) of the reconstructed 3D meshes from fusion of depth maps obtained from PSMNet (Chang and Chen, 2018) and our refined network. We obtain a RMSE of 1.3 cm on the simpler “Sofa and cushions” dataset. Note that our method captured high frequency structural details on the cushions which were not captured by PSMNet or the structured light sensor. “Plants and couch” represents a more difficult scene as it contained a directed light source casting shadows. For this dataset StereoDRNet obtained 2.1 cm RMSE whereas PSMNet obtained 2.5 cm RMSE. Notice, that our reconstruction is not only cleaner but produces minimal errors in the shadowed areas (shadows cast by book shelf and left plant). “Kitchen and bike” scene is cluttered and contains reflective objects making it the hardest dataset. While our system still achieved 2.1 cm RMSE, the performance of PSMNet degraded to 2.8 cm RMSE. Notice, that our reconstruction contains the faucet (highlighted by yellow box) in contrast to the structured light sensor and PSMNet reconstructions. For all evalu-

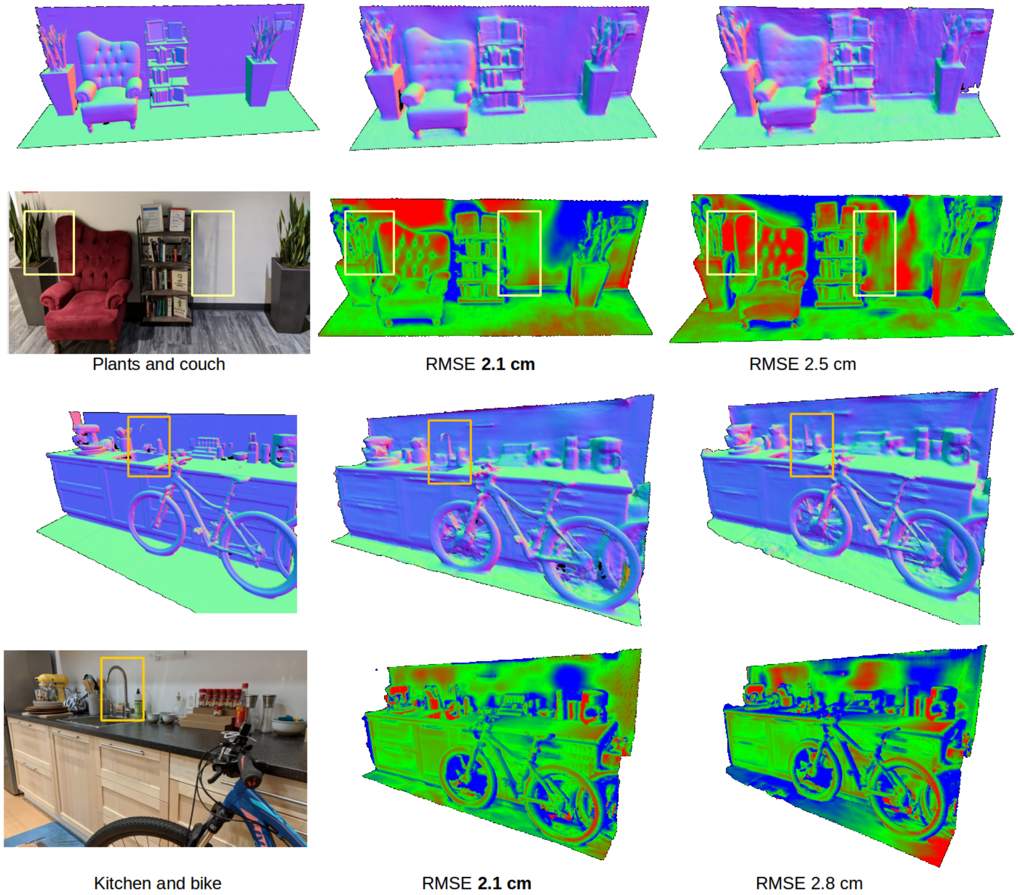


Figure 4.7: Comparison of 3D reconstruction using fusion of depth maps from our StereoDRNet network (middle), PSMNet (Chang and Chen, 2018) (right) and depth maps from the structured light system (left) described in (Whelan et al., 2018) (termed Ground Truth). We report and visualize point-to-plane distance RMS error on the reconstructed meshes with respect to the ground truth mesh. Dark yellow boxes represent the regions where our reconstruction yields details that the structured light sensor or PSMNet were not able to capture. Light yellow boxes represent regions where StereoDRNet outperforms PSMNet.

ations we used exactly same datasets for training and fine-tuning both PSMNet and our method StereoDRNet. We used synthetic dataset for training as described in Sec 3.6.1 and real-scene data for fine-tuning as described above in this section.

### 4.2.1 Comparison with MVS

We further compared our proposed 3D reconstruction pipeline, StereoDRNet with the state-of-the-art Multi-View Stereo Algorithm COLMAP (Schönberger et al., 2016) and learning based multi-view stereo algorithm MVSNet (Yao et al., 2018) in Table 4.1. The comparison is made on

| COLMAP | MVSNet | PSMNet | StereoDRNet(Ours) |
|---|---|---|---|
| 2.2 cm | 2.0 cm | 1.8 cm | **1.3** cm |

Table 4.1: This table reports RMSE in 3D Reconstruction task of the proposed Stereo-DRNet algorithm with the state-of-the-art methods in categories including traditional multi-view stereo COLMAP (Schönberger et al., 2016), learning based multi-view stereo MVSNet (Yao et al., 2018) and learning based two-view stereo PSMNet (Chang and Chen, 2018). The comparison is made on the "Sofa and cushions" scene as shown in Fig. 4.2

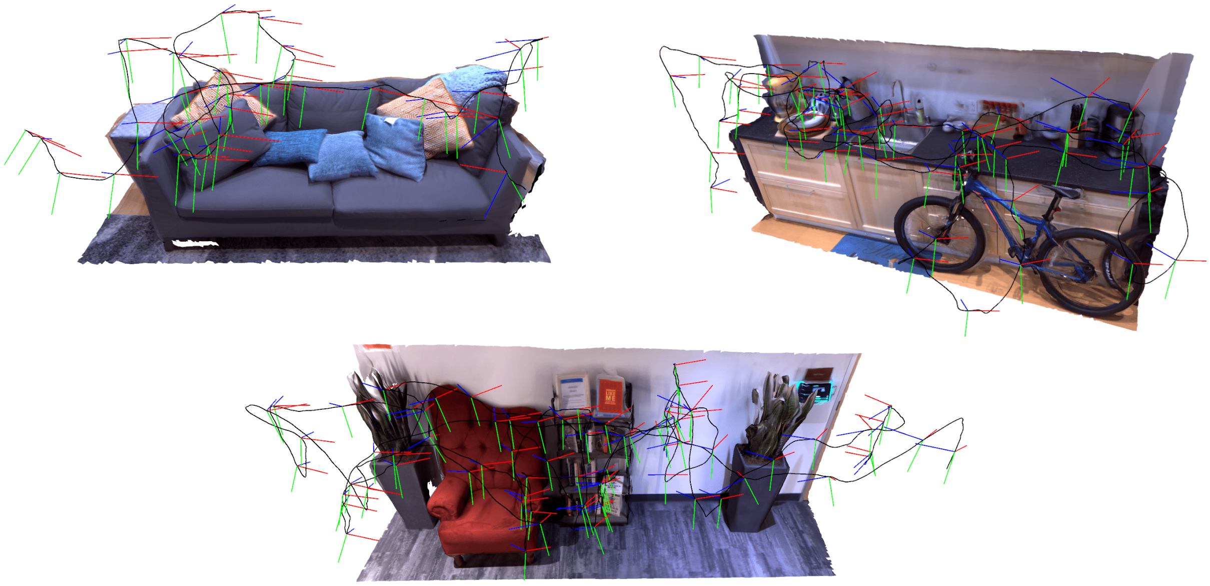


Figure 4.8: This figure shows the textured 3D reconstructions of "Sofa and cushions", "Plants and couch" and "kitchen and bike" scenes developed using KinectFusion (Newcombe et al., 2011a; Whelan et al., 2018) of depth maps generated form StereoSDRNet with refinement. We visualize the camera trajectory, from which the stereo images were taken, via a black curve. Note that for clarity we visualize every 30th frame used by the fusion system.

the "Sofa and cushions" scene as shown in Fig. 4.2. Note, that although MVSNet uses five views to predict one depth map, still the result is not as accurate as learning based two-view stereo algorithms as PSMNet (Chang and Chen, 2018) and our proposed method StereoDRNet. This shows that end-to-end multi-view stereo depth estimation learning is harder than end-to-end two fixed-views stereo depth estimation.

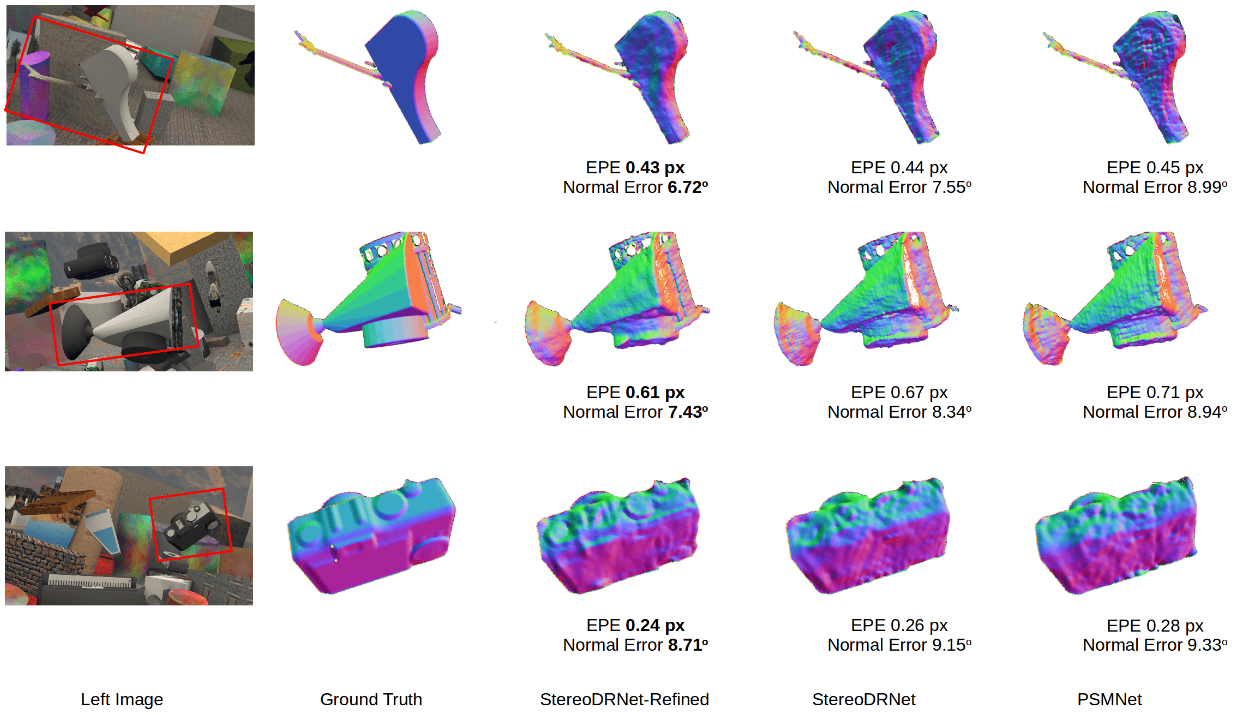


Figure 4.9: This figure demonstrates the surface normal visualizations of some objects (labeled with red boxes) reconstructed using **single** disparity map from SceneFlow dataset. We report EPE in disparity space and surface normal error in degrees. Notice, our refinement network improves the overall structure of the objects and makes them geometrically consistent.

### 4.2.2 3D Reconstruction Details

We show the textured 3D reconstructions of our indoor scene dataset in Fig. 4.8. Note that we used KinectFusion (Newcombe et al., 2011a) to fuse the depth maps into 3D spatial maps. We did not use any structure-from-motion (SfM) or external localization method for estimating camera trajectories. Hence, the camera views visualized in Fig. 4.8 are the output of the ICP (iterative closest point) procedure used by the KinectFusion (Newcombe et al., 2011a) system. We used manual adjustment followed by ICP to align the 3D reconstructions wherever necessary for our evaluations.

## 4.3 Effect of Refinement

Our refinement procedure not only improves the overall disparity error but also makes the prediction geometrically consistent. We calculate surface normal maps from disparity/depth

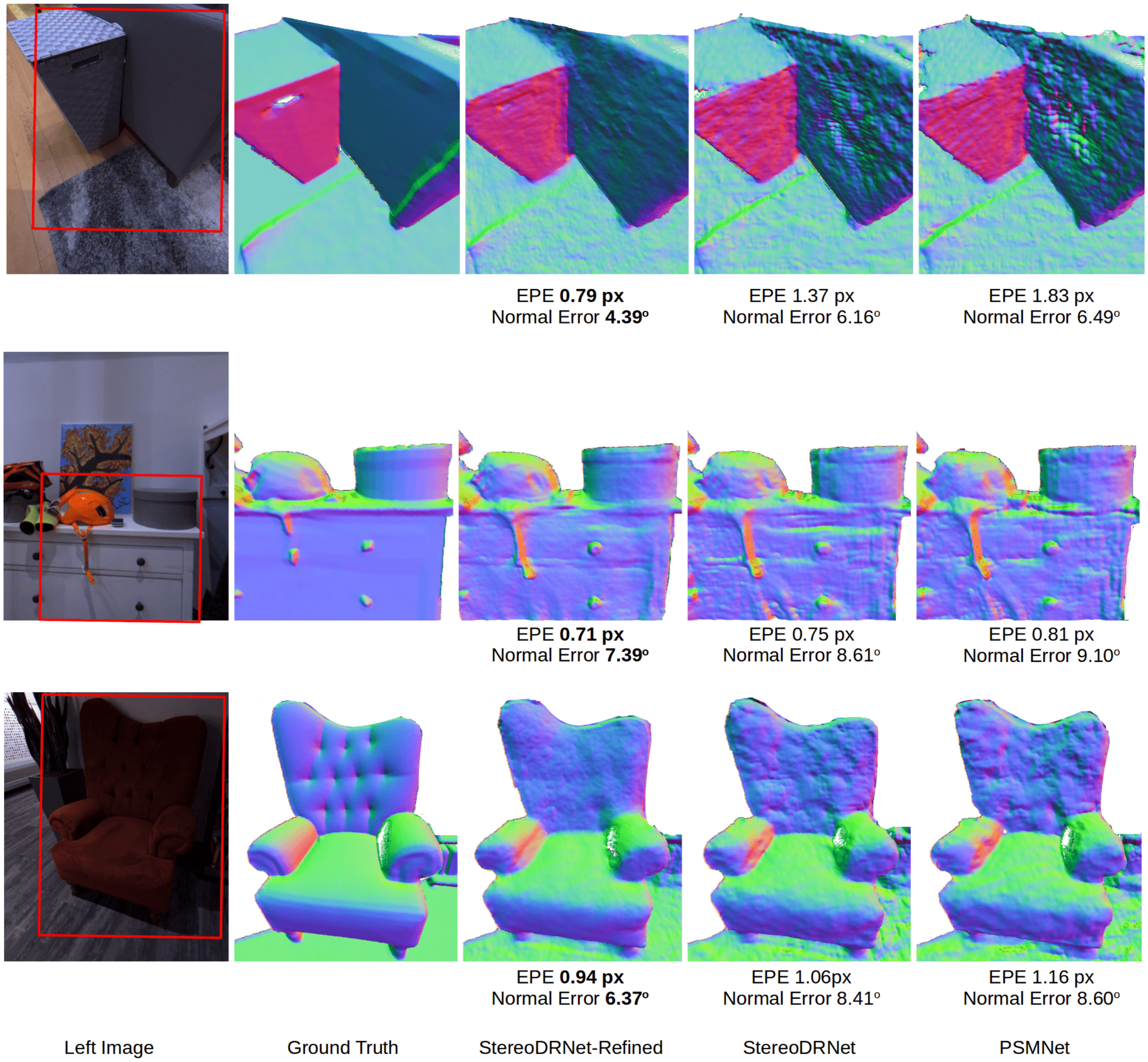


Figure 4.10: This figure shows the surface normal visualizations of some objects (labeled with red boxes) reconstructed using a **single** disparity map from our real dataset. We report EPE in disparity space and surface normal error in degrees. Notice that our refinement network improves the overall structure of the objects and makes them geometrically consistent.

maps using the approach described in KinectFusion (Newcombe et al., 2011a). We use a surface normal error metric to measure consistency in the disparity predictions (first order derivative). Figures 4.9 and 4.10 visualize how our refinement procedure improves the overall structure of objects. In some cases such as in the first comparison in Fig. 4.9 we observe little improvement in disparity prediction but large improvement in surface normals. Figure 4.10 demonstrates real scene disparity and derived surface normal predictions and proves that our refinement procedure works well on real world data in presence of shadows and dark lighting conditions. Dense 3D reconstruction methods such as KinectFusion (Newcombe et al., 2011a) use surface normals to calculate fusion parameters and confidence weights, hence it is important to predict geometrically consistent disparity or normal maps for high quality 3D reconstruction.

## 4.4 Conclusion

In this chapter we discussed how depth estimation from passive stereo images can be used to build 3D reconstruction of challenging scenes with homogeneous textured regions and surfaces with shadows and specular reflections. Proposed refinement network produces geometrically consistent disparity maps with the help of occlusion and view consistency cues. The use of perfect synthetic data and careful filtering of real training data enabled us to recover thin structures and sharp object boundaries. We demonstrate that our passive stereo system produces 3D reconstructions better than many state-of-the-art multi-view 3D reconstruction systems and approaches the quality of state-of-the-art structured light systems (Whelan et al., 2018).

# CHAPTER 5: LEARNING LOCAL SDF PRIORS FOR DETAILED 3D RECONSTRUCTION

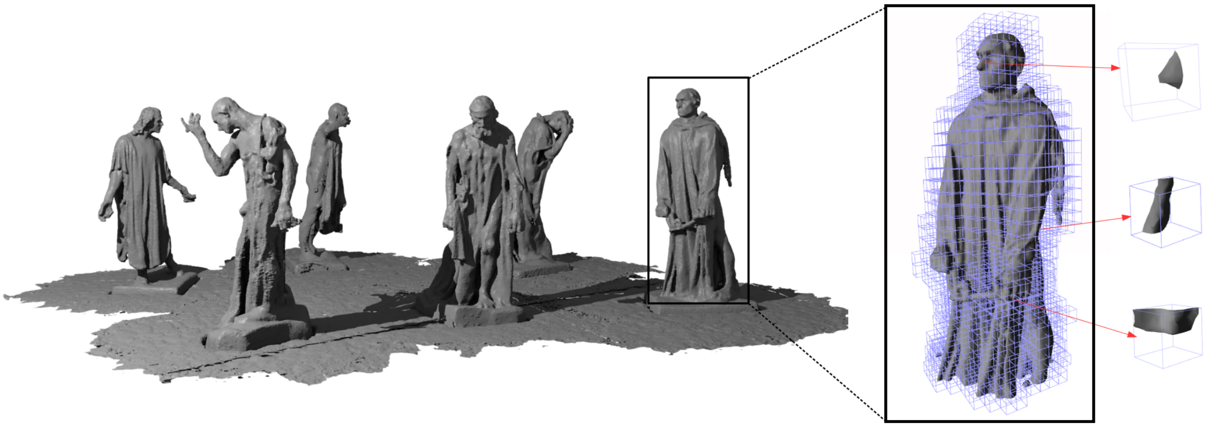


Figure 5.1: Reconstruction performed by our Deep Local Shapes (DeepLS) of the Burghers of Calais scene [Zhou and Koltun (2013)]. DeepLS represents surface geometry as a sparse set of local latent codes in a voxel grid, as shown on the right. Each code compresses a local volumetric SDF function, which is reconstructed by an implicit neural network decoder.

## 5.1 Introduction

A signed distance function (SDF) represents three-dimensional surfaces as the zero-level set of a continuous scalar field. This representation has been used by many classical methods to represent and optimize geometry based on raw sensor observations (Curless and Levoy, 1996; Klein and Murray, 2007; Stühmer et al., 2010; Newcombe and Davison, 2010; Newcombe et al., 2011a). In a typical use case, an SDF is approximated by storing values on a regularly-spaced voxel grid and computing intermediate values using linear interpolation. Depth observations can then be used to infer these values and a series of such observations are combined to infer the most likely SDF using a process called fusion.

Voxelized SDFs have been widely adopted and used successfully in a number of applications, but they have some fundamental limitations. First, the dense voxel representation requires significant amounts of memory (typically on a resource-constrained parallel computing device), which imposes constraints on resolution and the spatial extent that can be represented. These limits on resolution, as well as sensor limitations, typically lead to surface estimates that are missing thin structures and fine surface details. Second, as a non-parametric representation, SDF fusion can only infer surfaces that have been directly observed. Some surfaces are difficult or impossible for a typical range sensor to capture, and observing every surface in a typical environment is a challenging task. As a result, reconstructions produced by SDF fusion are often incomplete.

Recently, deep neural networks have been explored as an alternative representation for signed distance functions. According to the universal approximation theorem (Hornik et al., 1989), a neural network can be used to approximate any continuous function, including signed distance functions (Mescheder et al., 2019; Park et al., 2019; Michalkiewicz et al., 2019; Chen and Zhang, 2019). With such models, the level of detail that can be represented is limited only by the capacity and architecture of the network. In addition, a neural network can be made to represent not a single surface but a family of surfaces by, for example, conditioning the function on a latent code. Such a network can then be used as a parametric model capable of estimating the most likely surface given only partial noisy observations. Incorporating shape priors in this way allows us to move from the maximum likelihood (ML) estimation of classical reconstruction techniques to potentially more robust reconstruction via maximum a posteriori (MAP) inference.

These neural network representations have their own limitations, however. Most of the prior work on learning SDFs is object-centric and does not trivially scale to the detail required for scene-level representations. This is likely due to the global co-dependence of the SDF values at any two locations in space, which are computed using a shared network and a shared parameterization. Furthermore, while the ability of these networks to learn distributions over classes of shapes allows for robust completion of novel instances from known classes, it does not easily generalize to novel classes or objects, which would be necessary for applications in scene reconstruction. In scanned real-world scenes, the diversity of objects and object setups is usually too high to be covered by an object-centric training data distribution.

**Contribution.** In this work, we introduce Deep Local Shapes (DeepLS) to combine the benefits of both worlds, exposing a trade-off between the prior-based MAP inference of memory efficient deep global representations (e.g., DeepSDF), and the detail preservation of computationally efficient, explicit volumetric SDFs. We divide space into a regular grid of voxels, each with a small latent code representing signed distance functions in local coordinate frames and making use of learned local shape priors. These voxels can be larger than is typical in fusion systems without sacrificing on the level of surface detail that can be represented (c.f. Sec. 5.7.2.1), increasing memory efficiency. The proposed representation has several favorable properties, which are verified in our evaluations on several types of input data:

1. It relies on readily available local shape patches as training data and generalizes to a large variety of shapes,
2. provides significantly finer reconstruction and orders of magnitude faster inference than global, object-centric methods like DeepSDF, and
3. outperforms existing approaches in dense 3D reconstruction from partial observations, showing thin details with significantly better surface completion and high compression.

## 5.2 Related Work

The key contribution of this chapter is the application of learned local shape priors for reconstruction of 3D surfaces. This section will therefore discuss related work on traditional representations for surface reconstruction, learned shape representations, and local shape priors.

### 5.2.1 Traditional Shape Representations

Traditionally, scene representation methods can broadly be categorized into two categories, namely local and global approaches.

**Local approaches.** Most implicit surface representations from unorganized point sets are based on Blinn's idea of blending local implicit primitives (Blinn, 1982). (Hoppe et al., 1992) explicitly defined implicit surfaces by the tangent of the normals of the input points. (Ohtake et al., 2005) established more control over the local shape functions using quadratic surface fitting and blended these in a multi-scale partition of unity scheme. (Curless and Levoy, 1996) introduced volumetric integration of scalar approximations of implicit SDFs in regular grids. This technique was further extended into real-time systems (Klein and Murray, 2007; Stühmer et al., 2010; Newcombe and Davison, 2010; Newcombe et al., 2011a). Surfaces are also shown to be represented by surfels, i.e. oriented planar surface patches (Pfister et al., 2000; Keller et al., 2013; Whelan et al., 2015).

**Global approaches.** Global implicit function approximation methods aim to approximate single continuous signed distance functions using, for example, kernel-based techniques (Carr et al., 2001; Kazhdan et al., 2006; Ummenhofer and Brox, 2015; Fuhrmann and Goesele, 2014). Visibility or free space methods estimate which subset of 3D space is occupied, often by subdividing space into distinct tetrahedra (Labatut et al., 2009; Jancosek and Pajdla, 2011; Aroudj et al., 2017). These methods aim to solve for a globally view consistent surface representation.

Our work falls into the local surface representation category. It is related to the partition of unity approach (Ohtake et al., 2005), however, instead of using quadratic functions as local shapes, we use data-driven local priors to approximate implicit SDFs, which are robust to noise and can locally complete supported surfaces. While we also experimented with partition of unity blending of neighboring local shapes, we found it to be not required in practice, since our training formulation already includes border consistency (c.f. Sec 5.4.1), thus saving function evaluations during decoding. In comparison to volumetric SDF integration methods, such as SDF Fusion (Newcombe et al., 2011a), our approach provides better shape completion and denoising, while at the same time uses less memory to store the representation. Unlike point- or surfel-based methods, our method leads to smooth and connected surfaces.

### 5.2.2 Learned Shape Representations

Recently there has been lot of work on 3D shape learning using deep neural networks. This class of work can also be classified into four categories: point-based methods, mesh-based methods, voxel-based methods and continuous implicit function-based methods.

**Points.** The methods use generative point cloud models for scene representation (Achlioptas et al., 2017; Yang et al., 2017; Yuan et al., 2018). Typically, a neural network is trained to directly regress 3D coordinates of points in the point cloud.

**Voxels.** These methods provide non-parametric shape representation using 3D voxel grids which store either occupancy (Wu et al., 2015; Choy et al., 2016) or SDF information (Dai et al., 2017; Stutz and Geiger, 2018; Liao et al., 2018), similarly to the traditional techniques discussed above. These methods thus inherit the limitations of traditional voxel representations with respect to high memory requirements. Octree-based methods (Tatarchenko et al., 2017; Riegler et al., 2017; Häne et al., 2017) relax the compute and memory limitations of dense voxel methods to some degree and have been shown on voxel resolutions of up to $512^3$.

**Meshes.** These methods use existing (Sinha et al., 2016) or learned (Groueix et al., 2018; Ben-Hamu et al., 2018) parameterization techniques to describe 3D surfaces by morphing 2D planes. When using mesh representations, there is a tradeoff between the ability to support arbitrary topology and the ability to reconstruct smooth and connected surfaces. Works such as (Sinha et al., 2016; Ben-Hamu et al., 2018) are variations on deforming a sphere into more complex 3D shape, which produces smooth and connected shapes but limits the topology to shapes that are homeomorphic to the sphere. AtlasNet, on the other hand, warps multiple 2D planes into 3D which together form a shape of arbitrary topology, but this results in disconnected surfaces. Other works, such as Scan2Mesh (Dai and Nießner, 2019) and Mesh-RCNN(Gkioxari et al., 2019), use deep networks to predict meshes corresponding to range scans or RGB images, respectively.

**Implicit Functions.** Very recently, there has been significant work on learning continuous implicit functions for shape representations. Occupancy Networks (Mescheder et al., 2019) and PiFU (Saito et al., 2019) represent shapes using continuous indicator functions which specify which subset of 3D space the shapes occupy. Similarly, DeepSDF (Park et al., 2019) approximates shapes using Signed Distance Fields. We adopt the DeepSDF model as the backbone architecture for our local shape network.

Much of the work in this area has focused on learning object-level representations. This is especially useful when given partial observations of a known class, as the learned priors can often complete the shape with surprising accuracy. However, this also introduces three key difficulties. First, the object-level context means that generalization will be limited by the extent of the training set – objects outside of the training distribution may not be well reconstructed. Finally, object-level methods do not trivially scale to full scenes composed of many objects as well as surfaces (e.g. walls and floors). In contrast, DeepLS maintains separate representations for small, distinct regions of space, which allows it to scale easily. Furthermore, the local representation makes it easier to compile a representative training set; at a small enough scale most surfaces have similar structure.

### 5.2.3 Local Shape Priors

In early work on using local shape priors, (Gal et al., 2007) used a database of local surface patches to match partial shape observations. However, the ability to match general observations was limited by the size of the database as the patches could not be interpolated.(Ricao Canelhas et al., 2017) used both PCA and a learned autoencoder to map SDF subvolumes to lower-dimensional representations, approaching local shape priors from the perspective of compression. With this approach the SDF must be computed by fusion first, which serves as an information bottleneck limiting the ability to develop priors over fine-grained structures. In another work, (Xu et al., 2019) developed an object-level learned shape representation using a network that maps from images to SDFs .

This representation is conditioned on and therefore not independent of the observed image. (Williams et al., 2019) showed recently that a deep network can be used to fit a representation of a surface by training and evaluating on the same point cloud, using a local chart for each point which is then combined to form a surface atlas.

Their results are on complete point clouds in which the task is simply to densify and denoise, whereas we also show that our priors can locally complete surfaces that were not observed.

Other work on object-level shape representation has explored representations in which shapes are composed of smaller parts. Structured implicit functions used anisotropic Gaussian kernels to compose global implicit shape representations (Genova et al., 2019b). Similarly, CvxNets compose shapes using a collection of convex subshapes (Deng et al., 2019). Like ours, both of these methods show the promise of compositional shape modelling, but surface detail was limited by the models used. Last, concurrent work of (Genova et al., 2019a) combines a set of irregularly positioned implicit functions to improve details in full object reconstruction.

### 5.3 Review of DeepSDF

We will briefly review DeepSDF (Park et al., 2019). Let $f_\theta(\mathbf{x}, \mathbf{z})$ be a signed surface distance function modeled as a fully-connected neural network with trainable parameters $\theta$ and shape code $\mathbf{z}$. Then a shape $\mathcal{S}$ is defined as the zero level set of $f_\theta(\mathbf{x}, \mathbf{z})$:

$$\mathcal{S} = \left\{ \mathbf{x} \in \mathbb{R}^3 \mid f_\theta(\mathbf{x}, \mathbf{z}) = 0 \right\}. \tag{5.1}$$

In order to simultaneously train for a variety of shapes, a $\mathbf{z}$ is optimized for each shape while network parameters $\theta$ are shared for the whole set of shapes.

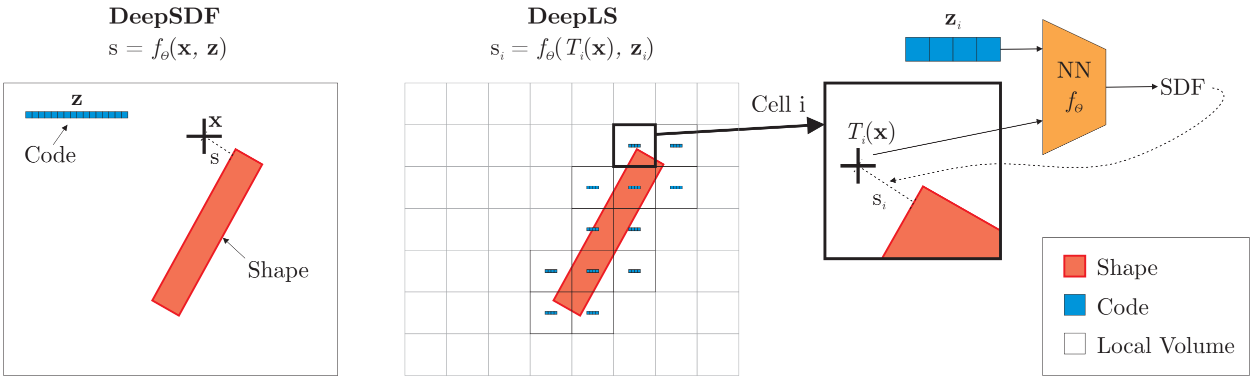


Figure 5.2: 2D example of DeepSDF (Park et al., 2019) and DeepLS (ours). DeepSDF provides global shape codes (left). We use the DeepSDF idea for local shape codes (center). Our approach requires a matrix of low-dimensional code vectors which in total require less storage than the global version. The gray codes are an indicator for empty space. The SDF to the surface is predicted using a fully-connected network that receives the local code and coordinates as input.

## 5.4 Deep Local Shapes

The key idea of DeepLS is to compose complex general shapes and scenes from a collection of simpler local shapes as depicted in Fig. 5.2. Scenes and shapes of arbitrary complexity cannot be described with a compact fixed length shape code such as used by DeepSDF. Instead it is more efficient and flexible to encode the space of smaller local shapes and to compose the global shape from an adaptable amount of local codes.

To describe a shape $\mathcal{S}$ defined over the space $\Omega$ using DeepLS, we first define a partition of the space into local volumes $\Omega_i \subseteq \Omega$ with associated local coordinate systems. Like in DeepSDF, but at a local level, we describe the surface in each local volume using a code $\mathbf{z}_i$. With the transformation $T_i(\mathbf{x})$ of the global location $\mathbf{x}$ into the local coordinate system, the global surface can be described as

$$\mathcal{S} = \left\{ \mathbf{x} \in \mathbb{R}^3 \mid \bigoplus_i w(\mathbf{x}, \Omega_i) f_\theta\left(T_i(\mathbf{x}), \mathbf{z}_i\right) = 0 \right\} , \tag{5.2}$$

where $w(\mathbf{x}, \Omega_i)$ weighs the contribution of the $i$th local shape to the global shape $\mathcal{S}$, $\bigoplus$ combines the contributions of local shapes, and $f_\theta$ is a shared autodecoder network for local shapes with trainable parameters $\theta$. Various ways of designing the combination operation and weighting function can be explored. From voxel-based tesselations of the space to more RBF-like point-based sampling to – in the limit – collapsing the volume of a local code into a point and thus making $\mathbf{z}_i$ a continuous function of the global space.

Here we focus the straight forward way of defining local shape codes over a sparsely allocated voxelization of the 3D space as illustrated in Fig. 5.2. We define $T_i(\mathbf{x})$ to transform a global point $\mathbf{x}$ into the local coordinate system of voxel cell $v_i$ by subtracting its center $\mathbf{x}_i$ from $\mathbf{x}$. The weighting function becomes the indicator function over the volume of voxel $v_i$.

Thus, DeepLS describes the global surface  as:

$$\mathcal{S} = \left\{\mathbf{x} \in \mathbb{R}^3 \mid \textstyle\sum_i \mathbb{1}_{\mathbf{x}\in v_i} f_\theta\left(T_i(\mathbf{x}), \mathbf{z}_i\right) = 0\right\} . \tag{5.3}$$

This can further be simplified to

$$\mathcal{S} = \left\{\mathbf{x} \in \mathbb{R}^3 \mid f_\theta\left(T_i(\mathbf{x}), z_i\right) = 0\,, i = \mathrm{idx}(\mathbf{x})\right\} , \tag{5.4}$$

where $\mathrm{idx}(\mathbf{x})$ determines the voxel index $i$ from the global position $\mathbf{x}$.

### 5.4.1 Shape Border Consistency

We found that with the proposed division of space (i.e. disjoint voxels for local shapes), training codes based only on samples within the voxel leads to inconsistent surface estimates at the voxel boundaries. One possible solution is to construct the surface as a partition of unity (Ohtake et al., 2005) with locally supporting basis functions to combine the decoded SDF values. We experimented with trilinear interpolation as an instance of this. However, such methods increase the number of required decoder evaluations to query an SDF value by a factor of eight.

Instead, we simply train the decoder weights and codes such that a local code can predict SDF values beyond the bounds of the voxel. In this regime we expect that the SDF values on the boundaries between voxels should be accurately computable from any of the abutting local codes.

We experimented with spheres (*i.e.* the $L_2$ norm) and voxels (*i.e.* the $L_\infty$ norm) for the extended indicator function and found that using an $L_\infty$ norm with a radius of $1.5$ times the voxel

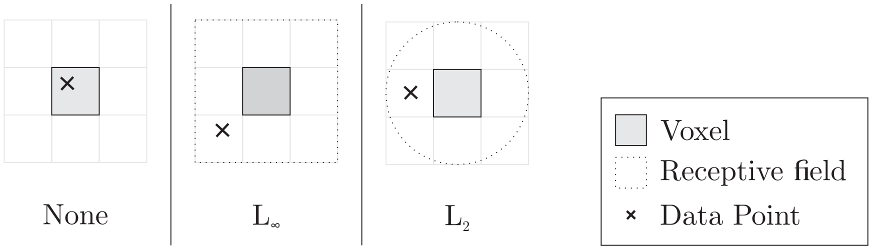


Figure 5.3: Square ($L_\infty$ norm) and spherical ($L_2$ norm) for the extended receptive fields for training local codes.

side-length provides accurate surfaces and is more efficient to compute than other alternatives (c.f. Figure 5.3). Our experiments show that the extended range is enough to effectively fight border artifacts (c.f. Sec. 5.7) while still providing the most efficient rendering.

### 5.4.2 Deep Local Shapes Training and Inference

Given a set of SDF pairs $\{(\mathbf{x}_j, s_j)\}_{j=1}^N$, sampled from a set of training shapes, we aim to optimize both the parameters $\theta$ of the shared shape decoder $f_\theta(\cdot)$ and all local shape codes $\{\mathbf{z}_i\}$ during training and only the codes during inference.

Let $\mathcal{X}_i = \{\mathbf{x}_j \mid L(T_i(\mathbf{x}_j)) < r\}$ denote the set of all training samples $\mathbf{x}_j$, falling within a radius $r$ of voxel $i$ with local code $\mathbf{z}_i$ under the distance metric $L$. Similar to DeepSDF, we train DeepLS by minimizing the negative log posterior over the training data $\mathcal{X}_i$:

$$\underset{\theta,\{\mathbf{z}_i\}}{\arg\min} \sum_i \sum_{\mathbf{x}_j \in_i} ||f_\theta\left(T_i(\mathbf{x}_j), \mathbf{z}_i\right) - s_j||_1 + \frac{1}{\sigma^2}||\mathbf{z}_i||_2^2 \,.$$

In order to encode a new scene or shape into a set of local codes, we fix decoder weights $\theta$ and find the maximum a-posteriori codes $\mathbf{z}_i$ as

$$\underset{\mathbf{z}_i}{\arg\min} \sum_{\mathbf{x}_j \in_i} ||f_\theta\left(T_i(\mathbf{x}_j), \mathbf{z}_i\right) - s_j||_1 + \frac{1}{\sigma^2}||\mathbf{z}_i||_2^2 \,, \tag{5.5}$$

given partial observation samples $\{(\mathbf{x}_j, s_j)\}_{j=1}^M$ with $\mathcal{X}_i$ defined as above.

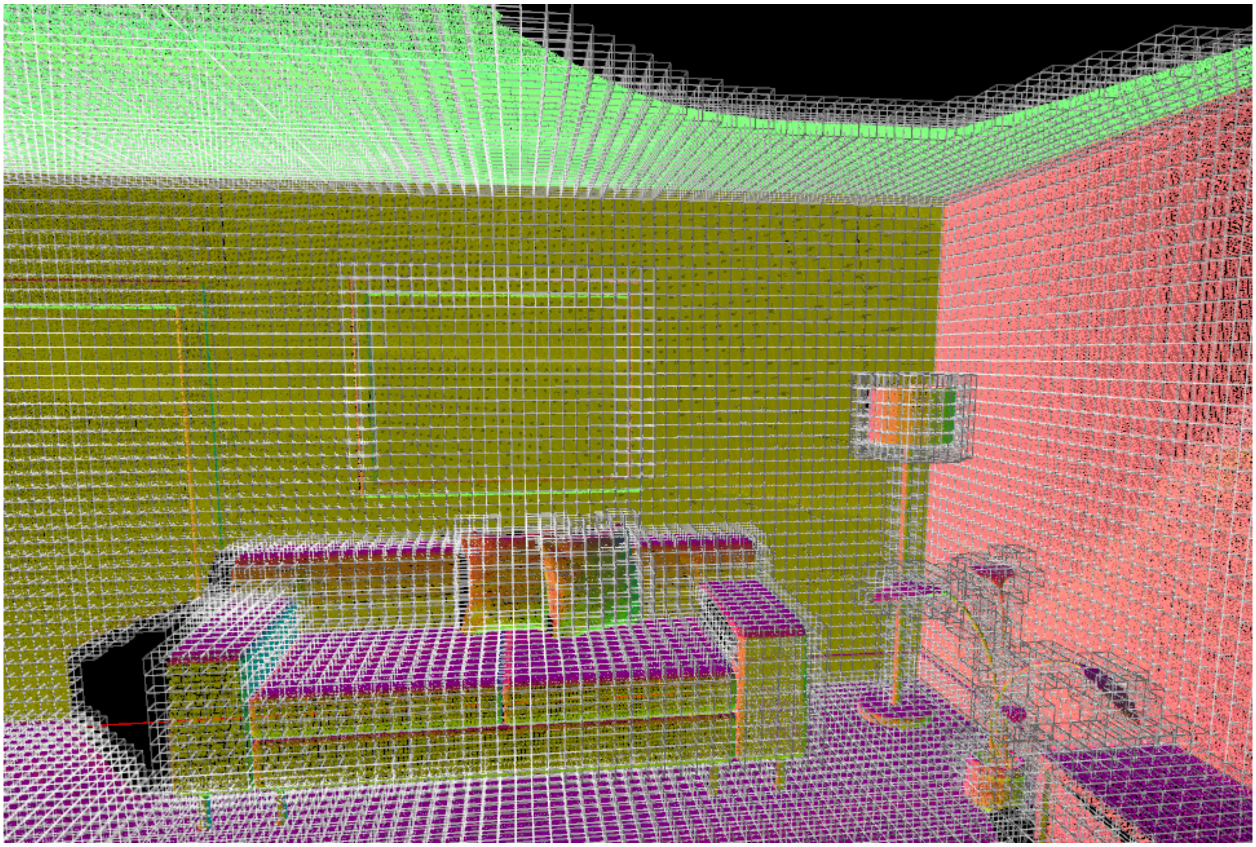


Figure 5.4: Instantiated local shape blocks in a scene. The blocks are allocated sparsely, based on available depth data, which makes the approach scale well to real world inputs.

### 5.5 Local Shape Space

In order to give a better intuition about the space of learned local priors, interpolation sequences between local surfaces are provided in Fig. 5.5. It should be noted that, in general, the space of possible functions in a voxel is much larger. Therefore, training local priors heavily restricts the space of solutions to those producing local SDF functions that describe reasonable surfaces. Additionally, Fig. 5.4 show all allocated blocks in a scene, which together reconstruct the whole surface.

#### 5.5.1 Sample Generation from Depth Maps

For sampling data pairs $(\mathbf{x}_j, s_j)$, we distinguish between sampling from meshes and depth observations. For meshes, the method proposed by (Park et al., 2019) is used. For depth obser-

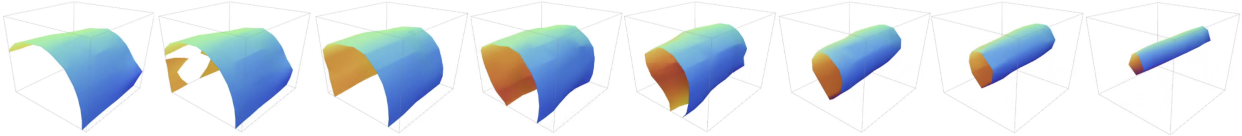


Figure 5.5: Interpolation in latent space of a local shape code between a flat surface and a pole.

vations, we estimate normals from the depth map and sample points in 3D that are displaced slightly along the normal direction, where the SDF value is assumed to be the magnitude of displacement. In addition to those samples, we obtain free space samples along the observation rays. Sample generation from depth scans consists of the following steps: (1) For a given scene, we generate a collection of 3D points from depth maps. (2) For each depth point, we create one sample with zero SDF, and several positive and negative SDF samples by moving the sample along the pre-computed surface normal by $1.5$ cm and $-1.5$ cm, respectively. The accompanying SDF value is chosen as the moved distance. (3) We generate additional free space samples along the observation rays as shown in Fig 5.6. Further, we weight each set of points inversely based on the depth of the initial scan point, to ensure that accurate points closer to the scanning device are weighted higher. This procedure is described in detail in TSDF Fusion (Newcombe et al., 2011a).

## 5.6 Experimental Setup

**Autodecoder Network** The DeepLS autodecoder network is a lighter version of the network proposed for DeepSDF (Park et al., 2019). It consists of four fully-connected layers, separated by leaky ReLUs and a tanh at the end, producing values in $[-1, 1]$ that are then scaled by the chosen SDF truncation value. Each layer has $128$ output neurons. Fig. 5.7a shows the result of a small study to find the best latent code size in a trade-off between accuracy and compression. We chose a latent size of $125$, leaving us with $128$ input neurons for the first network layer.

**Training** The output of the network is trained to produce truncated SDF values. To this end, tanh is also applied on the appropriately scaled ground truth SDF before computing the loss against the network output. We chose the scale so that the interval $[-0.9, 0.9]$ after tanh covers

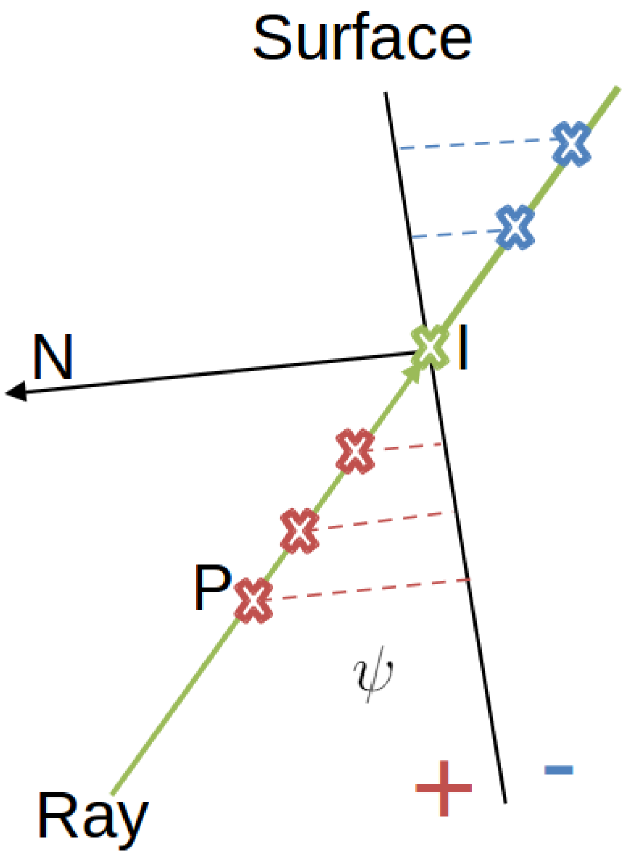
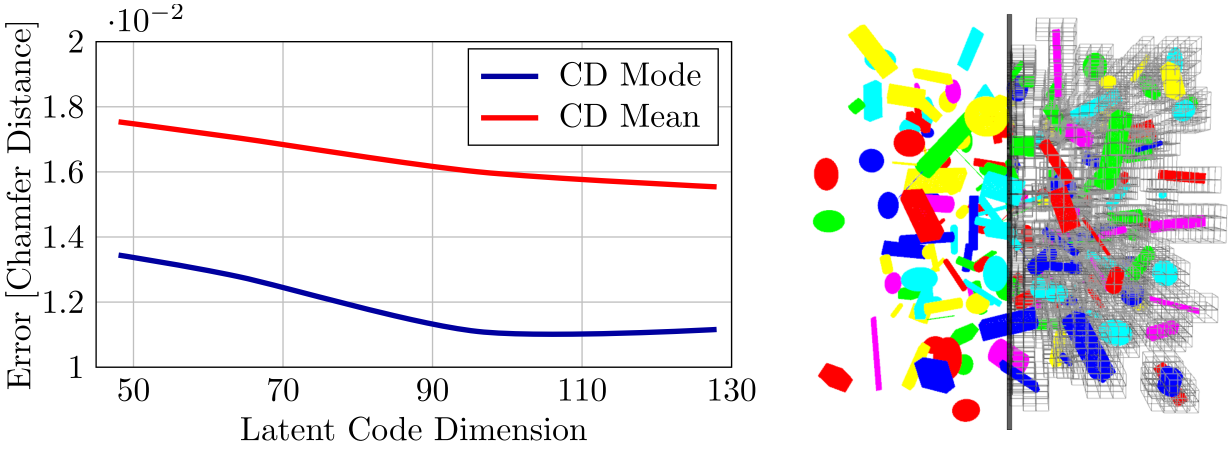


Figure 5.6: This figure demonstrates how positive and negative SDF samples can be generated for a particular depth observation.

approximately two blocks. We optimize codes and network parameters using the Adam optimizer with initial learning rate of $0.01$, which we decrease twice over the course of training.

**Training Data** The training data to learn local shape priors consists of three different categories of shapes. The first category contains simple primitive shapes, as shown in Fig. 5.7b, with random 6-DOF pose in space. The second category consists of ShapeNet (Chang et al., 2015) training meshes: We sampled a subset of $200$ models from each training set of the classes *airplane*, *chair*, *lamp*, *sofa*, and *table*. Each model was split into $32 \times 32 \times 32$ local shape blocks. The last category consists of examples of the Stanford 3D scanning repository (Sta, 1996), namely *bunny* and *dragon*.



(a) Latent code size

(b) Primitives for training

Figure 5.7: Fig. (a) shows the effect of changing the latent code dimensions on the Chamfer distance test error on airplanes class of ShapeNet (Chang et al., 2015). Fig. (b) shows an example for a scene containing 200 primitives shapes as used for training the local shape priors. On the right side, the instantiated local shape blocks are shown.

## 5.7 Experiments

The experiment section is structured as follows. First, we compare DeepLS against recent deep learning methods (e.g. DeepSDF, AtlasNet) in Sec. 5.7.1. Then, we present results for scene reconstruction and compare them against related approaches on both synthetic and real scenes in Sec. 5.7.2.3.

**Experiment setup.** The models used in the following experiments were trained on a set of local shape patches, obtained from 200 primitive shapes (e.g. cuboids and ellipsoids) and a total of 1000 shapes from the ShapeNet training set (200 each for the airplane, table, chair, lamp, and sofa classes). Our decoder is a four layer MLP, mapping from latent codes of size 128 to the SDF value.

### 5.7.1 Object Reconstruction

#### 5.7.1.1 ShapeNet (Chang et al., 2015)

We quantitatively evaluate surface reconstruction accuracy of DeepLS and other shape learning methods on various classes from the ShapeNet dataset. Quantitative results for the chamfer distance error are shown in Table 5.1. As can be seen DeepLS improves over related approaches by approximately one order of magnitude. It should be noted that this is not a comparison between equal methods since the other methods infer a global, object-level representation that comes with other advantages. Also, the parameter distribution varies significantly (c.f. Tab. 5.1). Nonetheless, it proves that local shapes lead to superior reconstruction quality and that implicit functions modeled by a deep neural network are capable of representing fine details. Qualitatively, DeepLS encodes and reconstructs much finer surface details as can be seen in Fig. 5.8.

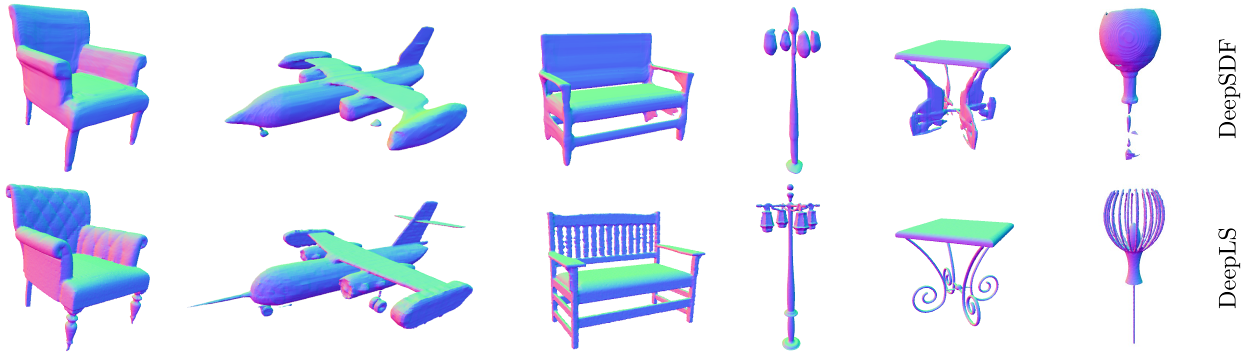


Figure 5.8: Qualitative comparison of DeepLS with DeepSDF on some shapes from the ShapeNetV2 dataset.

| Method | ShapeNet Category (Chamfer Dist. Error) chair | plane | table | lamp | sofa | Decoder Params | Represent. Params |
|---|---|---|---|---|---|---|---|
| AtlasNet-Sph. (Groueix et al., 2018) | 0.752 | 0.188 | 0.725 | 2.381 | 0.445 | 3.6 M | 1.0 K |
| AtlasNet-25 (Groueix et al., 2018) | 0.368 | 0.216 | 0.328 | 1.182 | 0.411 | 43.5 M | 1.0 K |
| DeepSDF (Park et al., 2019) | 0.204 | 0.143 | 0.553 | 0.832 | 0.132 | 1.8 M | **0.3 K** |
| DeepLS | **0.030** | **0.018** | **0.032** | **0.078** | **0.044** | **0.05 M** | 312 K |

Table 5.1: Comparison for reconstructing shapes from the ShapeNet test set, using the Chamfer distance. Note that due to the much smaller decoder, DeepLS is also orders of magnitudes faster in decoding (querying SDF values).

#### 5.7.1.2 Efficiency Evaluation on Stanford Bunny

Further, we show the superior inference efficiency of DeepLS with a simple experiment, illustrated in Figure 5.9. A DeepLS model was trained for just one minute on a single GPU on a dataset composed only of randomly oriented primitive shapes. It is used to infer local codes that pose an implicit representation of the Stanford Bunny, resulting in an RMSE of only 0.03% relative to the length of the diagonal of the minimal ground truth bounding box, highlighting the ability of DeepLS to generalize to novel shapes. For comparison, we also trained a DeepSDF model to represent only the Stanford Bunny (jointly training latent code and decoder). In order to achieve the same surface error, this model required over 8 days of GPU time, showing that the high compression rates and object-level completion capabilities of DeepSDF and related techniques comes at the cost of long training and inference times. This is likely caused at least in part by gradient computation amongst all training samples, which we avoid by subdividing physical space and optimizing local representations in parallel.

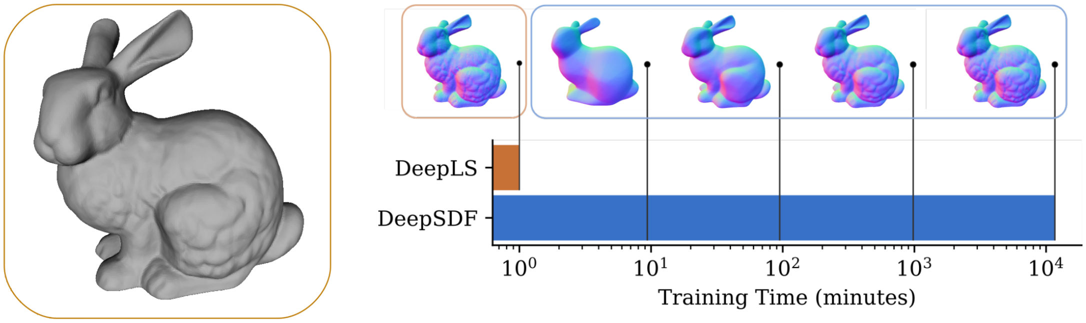


Figure 5.9: A comparison of the efficiency of DeepLS and DeepSDF. With DeepLS, a model trained for one minute is capable of reconstructing the Stanford Bunny in full detail. We then trained a DeepSDF model to represent the same signed distance function corresponding to the Stanford Bunny until it reaches the same accuracy. This took over 8 days of GPU time (note the log scale of the plot).

### 5.7.2 Scene Reconstruction

We evaluate the ability of DeepLS to reconstruct at scene scale using synthetic (in order to provide quantitative comparisons) and real depth scans. For synthetic scans, we use the ICL-NUIM RGBD benchmark dataset (Handa et al., 2014). The evaluation on real scans is done using

| Method | mean | kt0 | kt1 | kt2 | kt3 |
|---|---|---|---|---|---|
| TSDF Fusion | 5.42 mm | 5.35 mm | 5.88 mm | 5.17 mm | 5.27 mm |
| DeepLS | **4.92 mm** | **5.15 mm** | **5.48 mm** | **4.32 mm** | **4.71 mm** |

Table 5.2: Surface reconstruction accuracy of DeepLS and TSDF Fusion (Newcombe et al., 2011a) on the synthetic ICL-NUIM dataset (Handa et al., 2014) benchmark

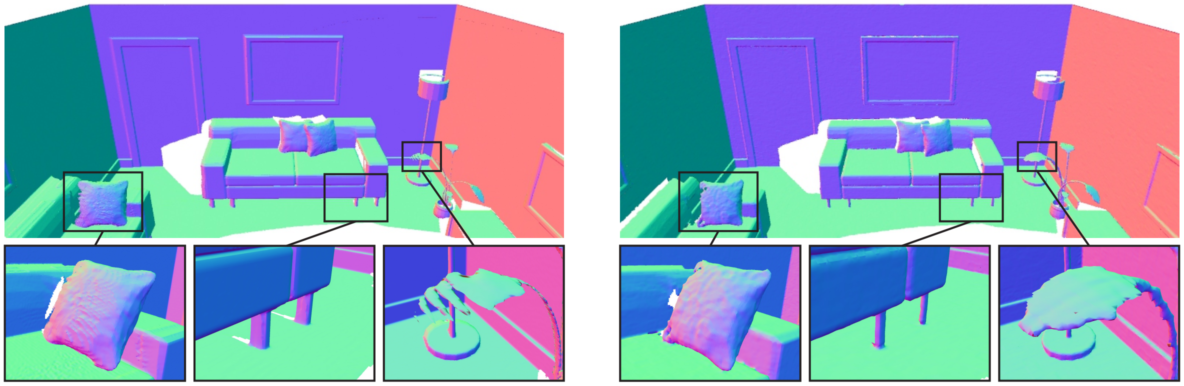
| Method | mean | kt0 | kt1 | kt2 | kt3 |
|---|---|---|---|---|---|
| KinectFusion | 5.42 mm | 5.35 mm | 5.88 mm | 5.17 mm | 5.27 mm |
| DeepLS | **4.92 mm** | **5.15 mm** | **5.48 mm** | **4.32 mm** | **4.71 mm** |

Table 5.3: Surface reconstruction accuracy of DeepLS and TSDF Fusion (Newcombe et al., 2011a) on the synthetic ICL-NUIM dataset (Handa et al., 2014) benchmark

the 3D Scene Dataset (Zhou and Koltun, 2013). For quantitative evaluation, the asymmetric Chamfer distance metric provided by the benchmark (Handa et al., 2014) is used.

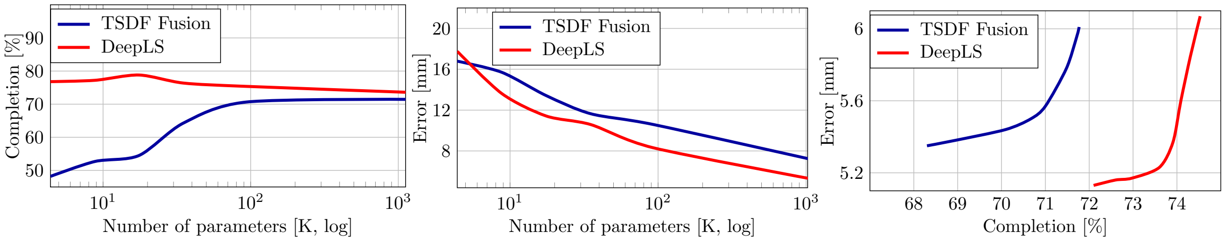
#### 5.7.2.1 Synthetic ICL-NUIM Dataset Evaluation

We provide quantitative measurements of surface reconstruction quality on all four ICL-NUIM sequences in Table 5.3, where each system has been tuned for lowest surface error. We also show results qualitatively in Fig. 5.10 and show additional results, e.g. on data with artificial noise, in Sec. 5.7.2.2. Most surface reconstruction techniques involve a tradeoff between surface accuracy and completeness. For TSDF fusion systems such as KinectFusion (Newcombe et al., 2011a), this tradeoff is driven by choosing a truncation distance and the minimum confidence at which surfaces are extracted by marching cubes. With DeepLS, we only extract surfaces up to some fixed distance from the nearest observed depth point, and this threshold is what trades off accuracy and completion of our system. For a full and fair comparison, we derived a pareto-optimal curve by varying these parameters for the two methods on the ‘kt0‘ sequence of the ICL-NUIM benchmark and plot the results in Figure 5.11. We measure completion by computing the fraction of ground truth points for which there is a reconstructed point within 7 $\mathrm{mm}$. Generally, DeepLS can reconstruct more complete surfaces at the same level of accuracy as SDF Fusion.



(a) TSDF Fusion (Newcombe et al., 2011a) (b) DeepLS (ours)

Figure 5.10: Qualitative results of TSDF Fusion (Newcombe et al., 2011a) (left) and DeepLS (right) for scene reconstruction on a synthetic ICL NUM scene. The highlighted areas indicate the ability of DeepLS to handle oblique viewing angles, partial observation, and thin structures.



(a) Completion (b) Surface error (c) Completion vs. error

Figure 5.11: Comparison of completion (a) and surface error (b) as a function of representation parameters on a synthetic scene from the ICL-NUIM (Handa et al., 2014) dataset. In contrast to TSDF Fusion, DeepLS maintains reconstruction completeness almost independent of compression rate. On the reconstructed surfaces (which is 50% less for TSDF Fusion) the surface error decreases for both methods (c.f. Fig. 5.12). Plot (c) shows the trend of surface error vs. mesh completion. DeepLS consistently shows higher completion at the same surface error. It scores less error than TSDF Fusion in all but the highest compression setting but it produces nearly two times more complete reconstruction than TSDF Fusion at this compression rate.

The number of representation parameters used by DeepLS is theoretically independent of the rendering resolution and only depends on the resolution of the local shapes. In contrast, traditional volumetric scene reconstruction methods such as TSDF Fusion have a tight coupling between number of parameters and the desired rendering resolution. We investigate the relationship between representation size per unit volume of DeepLS and TSDF Fusion by evaluating the surface error and completeness as a function of the number of parameters. As a starting point we choose a representation that uses $8^3$ parameters per 5.6cm $\times$ 5.6cm $\times$ 5.6cm volume (7 mm voxel resolution). To increase compression we increase the voxel size for TSDF Fusion and the local

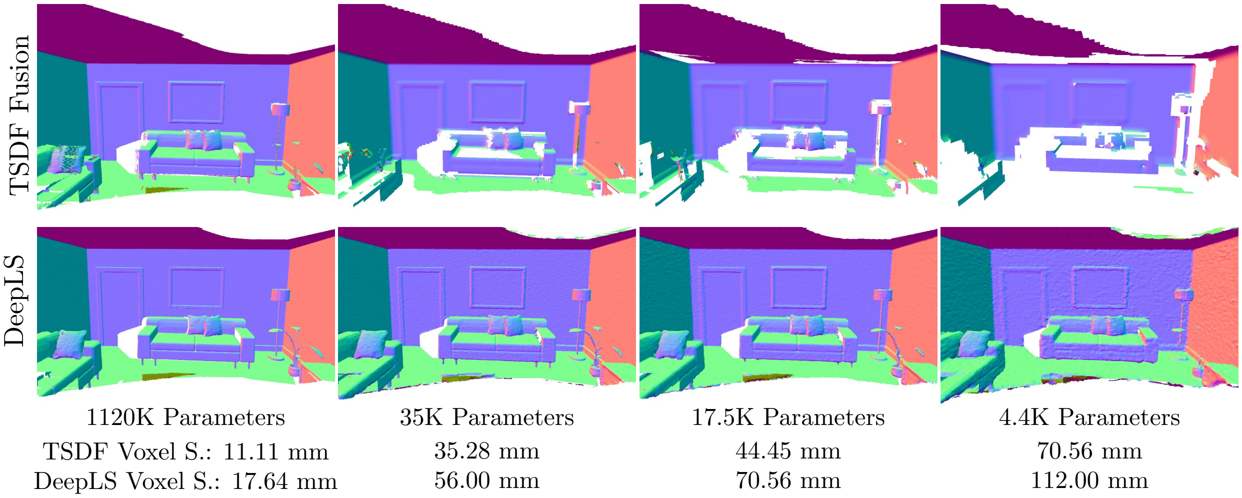
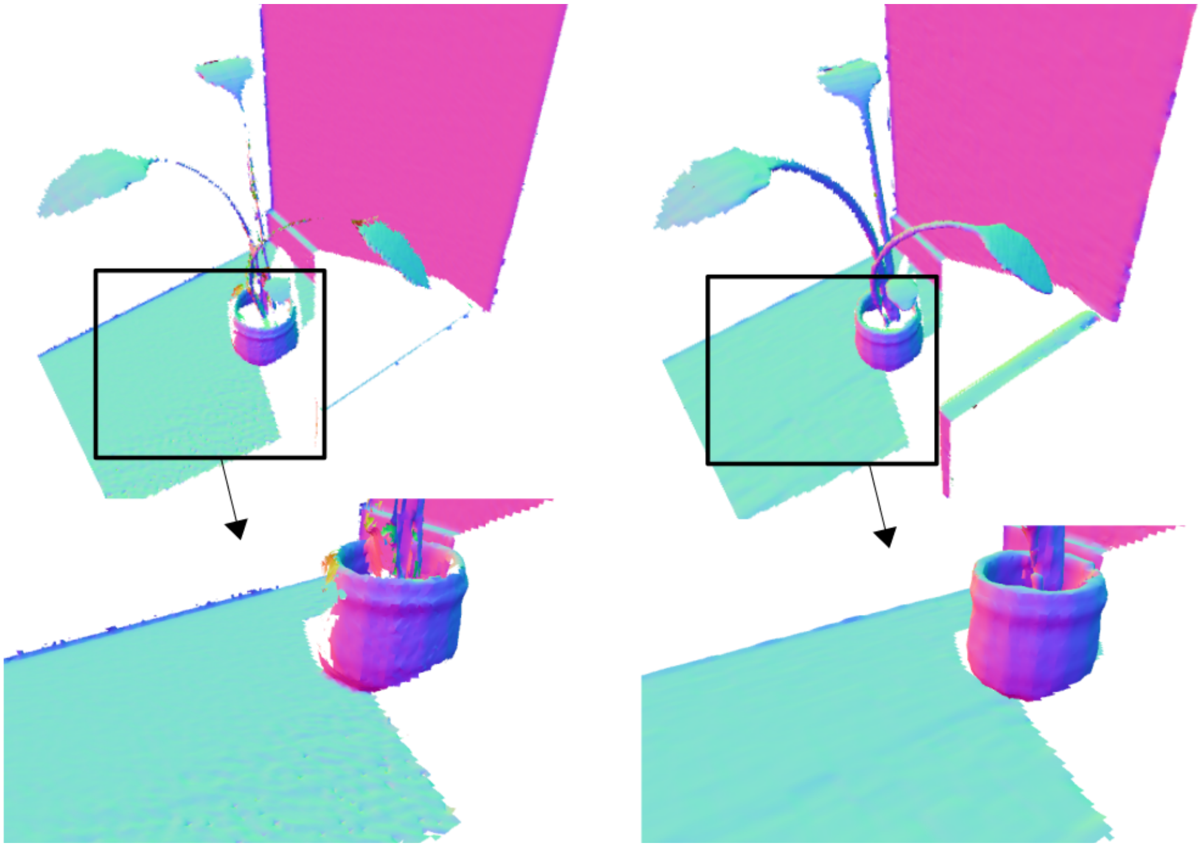


Figure 5.12: Qualitative analysis of representation size with DeepLS and TSDF Fusion (Newcombe et al., 2011a) on a synthetic scene in the ICL-NUIM (Handa et al., 2014) dataset. DeepLS is able to retain details at higher compression rates (lower number of parameters). It achieves these compression rates by using bigger local shape voxels, leading to a stronger influence of the priors.

| Method | Burghers | | Lounge | | CopyRoom | | StoneWall | | TotemPole | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Error | Comp | Error | Comp | Error | Comp | Error | Comp | Error | Comp |
| TSDF F. (Newcombe et al., 2011a) | 10.11 | 85.46 | 11.71 | 85.17 | 12.35 | 83.99 | 14.23 | 91.02 | 13.03 | 83.73 |
| DeepLS | **5.74** | **95.78** | **7.38** | **96.00** | **10.09** | **99.70** | **6.45** | **91.37** | **8.97** | **87.23** |

Table 5.4: Quantitative evaluation of DeepLS with TSDF Fusion on 3D Scene Dataset (Zhou and Koltun, 2013). The error is measured in mm and *Comp* (completion) corresponds to the percentage of ground truth surfaces that have reconstructed surfaces within 7 mm. Results suggest that DeepLS produces more accurate and complete 3D reconstruction in comparison to volumetric fusion methods on real depth acquisition datasets.

shape code volume size for DeepLS. We provide the quantitative and qualitative analysis of the scene reconstruction results with varying representation size in Fig. 5.11 (a and b) and Fig. 5.12 respectively. The plots in Fig. 5.11 show conclusively that TSDF Fusion drops to about 50% less complete reconstructions while DeepLS maintains completeness even at the highest compression rate, using only 4.4K parameters for the full scene. Quantitatively, TSDF Fusion also achieves low surface error for high compression. However, this can be contributed to the used ICL-NUIM benchmark metric, which does not strongly punish missing surfaces.



(a) TSDF Fusion

(b) DeepLS

Figure 5.13: The figure shows a part of the ICL-NUIM kt0 scene (Handa et al., 2014), reconstructed from samples with artitificial noise of $\sigma = 0.015$. DeepLS shows better denoising properties than TSDF Fusion. For the whole ICL-NUIM benchmark scene, DeepLS achieves a surface error of **6.41** mm with **71.04** % completion while TSDF Fusion has an error of 7.29 mm and 68.53 % completion.

#### 5.7.2.2 Comparisons for Synthetic Noise

Fig. 5.13 shows results of DeepLS and TSDF Fusion on an ICL-NUIM benchmark scene with artificial noise of $\sigma = 0.015$. The learned local shape priors of DeepLS effectively are able to find plausible surfaces given the noisy observations, which results in smoother surfaces in comparison to TSDF Fusion.

#### 5.7.2.3 Evaluation on Real Depth Scans

We evaluate DeepLS on the 3D Scene Dataset (Zhou and Koltun, 2013), which contains several scenes captured by commodity structured light sensors, and a challenging scan of thin objects. In order to also provide quantitative errors we assume the reconstruction performed by volumetric fusion (Newcombe et al., 2011a) of all depth frames to be the *ground truth*. We then apply DeepLS and TSDF fusion on a small subset of depth frames, taking every 10th frame in the capture sequence. The quantitative results of this comparison are detailed in Table 5.4 for various scenes. It is shown that DeepLS produces both more accurate and more complete 3D reconstructions. Furthermore, we provide qualitative examples of this experiment in Fig. 5.14 for the outdoor scene “Burghers of Calais” and in Fig. 5.15 for the indoor scene “Lounge”. Notice, that DeepLS preserves more details on the faces of the statues in “Burghers of Calais” scene and reconstructs thin details such as leaves of the plants in “Lounge” scene.

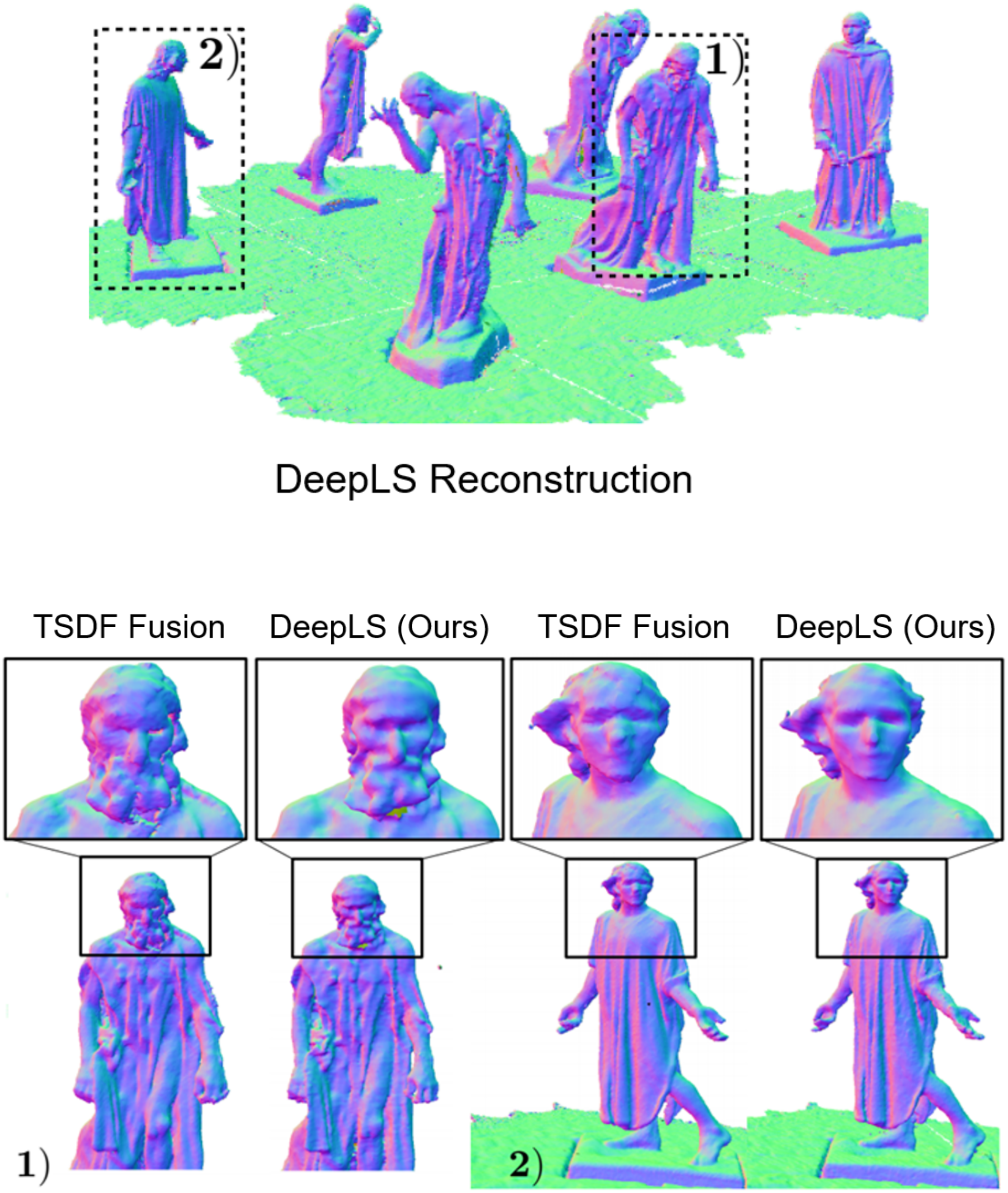


Figure 5.14: Qualitative results for DeepLS and TSDF Fusion (Newcombe et al., 2011a) on “Burghers of Calais scene” scenes of the 3D Scene Dataset (Zhou and Koltun, 2013).

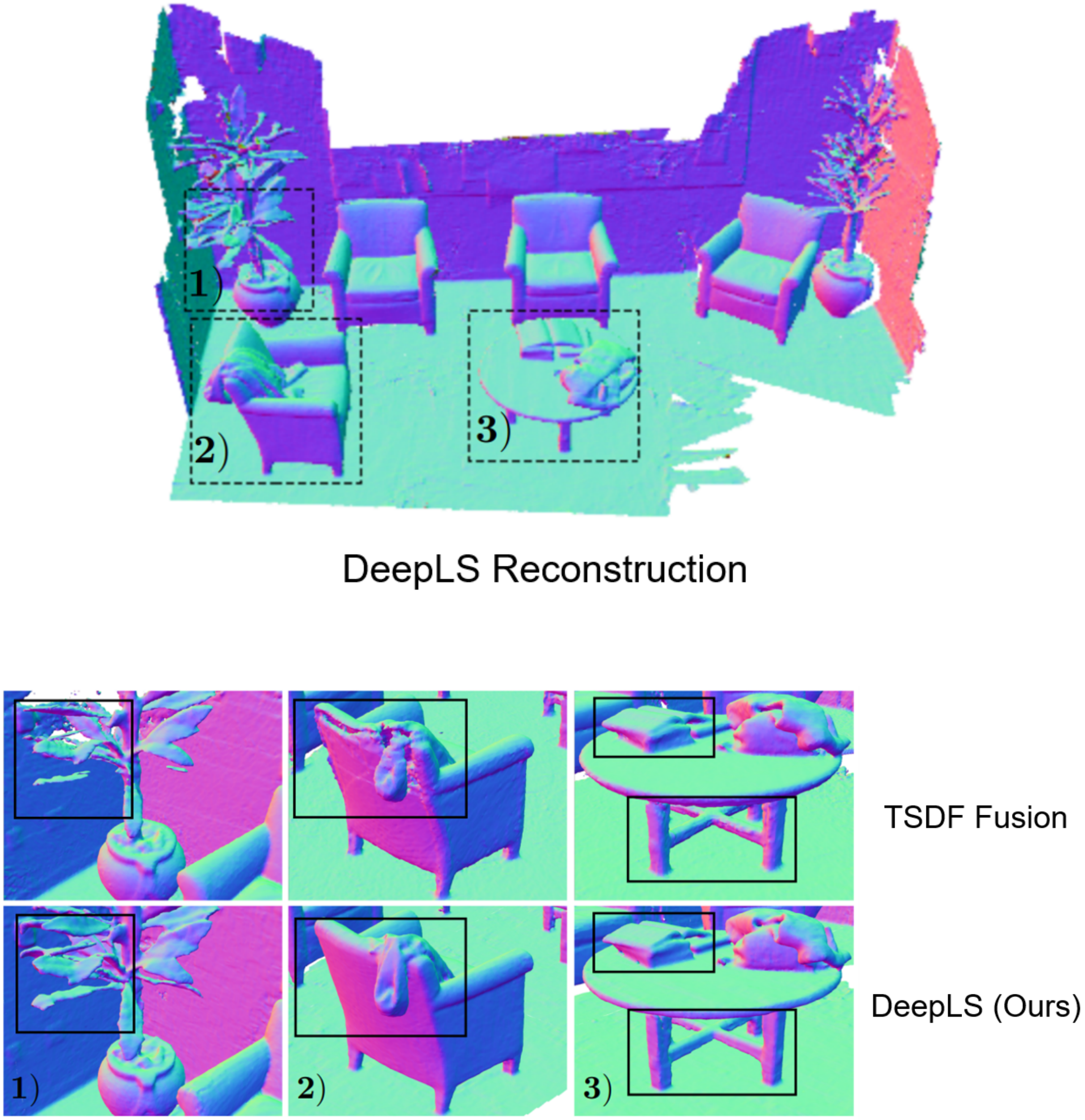
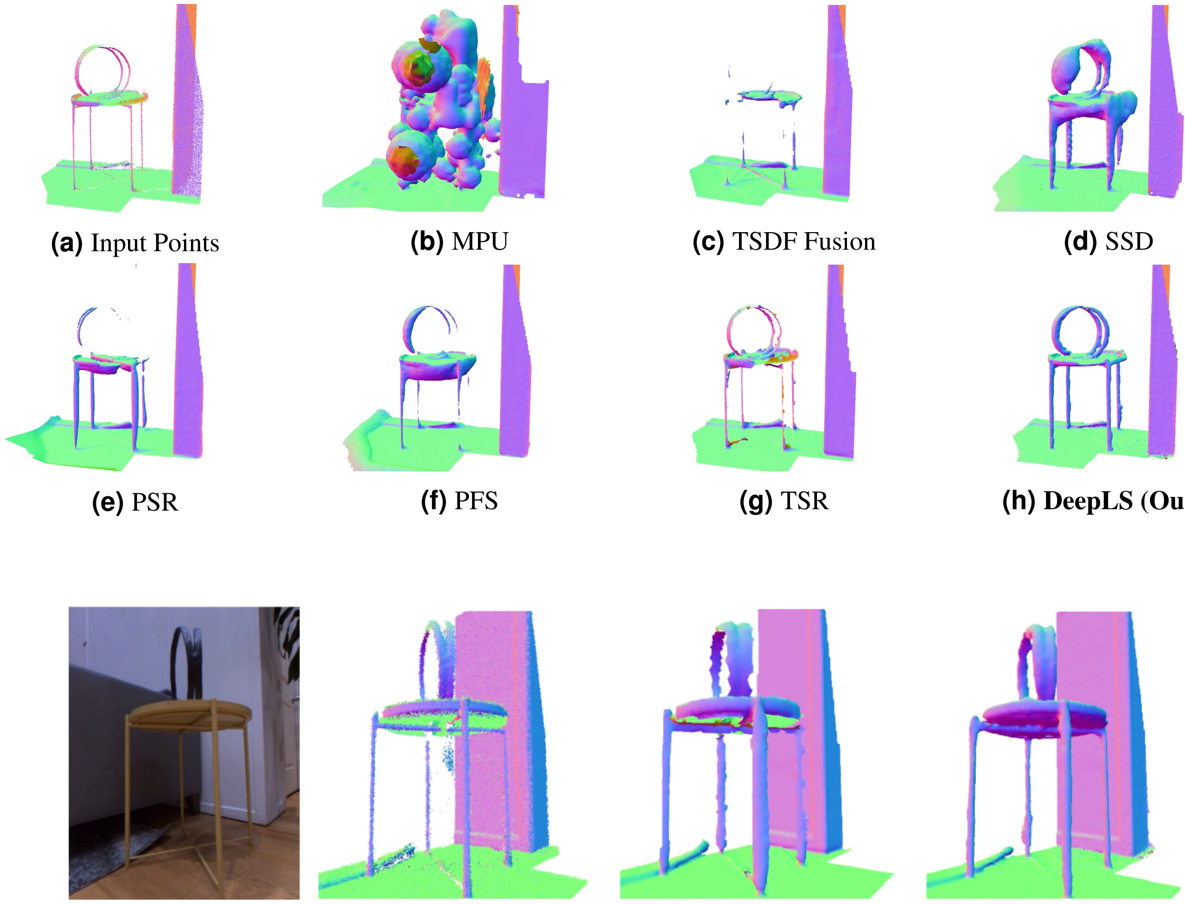


Figure 5.15: Qualitative results for DeepLS and TSDF Fusion (Newcombe et al., 2011a) on "Lounge" scenes of the 3D Scene Dataset (Zhou and Koltun, 2013).

#### 5.7.2.4 Evaluation on Depth Scans with Thin Structures (Straub et al., 2019a)

we specifically analyse the strength of DeepLS in representing and completing thin local geometry. We collected a scan from an object consisting of two thin circles kept on a stool with long but thin cylindrical legs Fig. 5.16. The 3D points were generated by a structured light sensor described in (Straub et al., 2019a) and used in  (Whelan et al., 2018; Chabra et al., 2019).SLAM system, similar to state-of-the-art systems like (Engel et al., 2017; Mur-Artal et al., 2015) was used to provide 6 degree of freedom (DoF) poses for individual depth frames form the sensor. The stool was scanned from limited directions leading to very sparse set of points on the stool's surface and legs. We compared our results on this dataset to several 3D methods including TSDF Fusion (Newcombe et al., 2011a), Multi-level Partition of Unity (MPU) (Ohtake et al., 2005), Smooth Signed Distance Function (Calakli and Taubin, 2011), Poisson Surface Reconstruction (Kazhdan and Hoppe, 2013), PFS (Ummenhofer and Brox, 2015) and TSR (Aroudj et al., 2017). We found that due to lack of points and thin surfaces most of the methods failed to either represent details or complete the model. MPU (Ohtake et al., 2005), which fits quadratic functions in local grids and is very related to our work, fails in this experiment (see Fig. 5.16b). This indicates that our learned shape priors are more robust than fixed parameterized functions. Methods such as PSR (Kazhdan and Hoppe, 2013), SSD (Calakli and Taubin, 2011) and PFS (Ummenhofer and Brox, 2015) fit global implicit function to represent shapes. These methods made the thin shapes thicker than they should be. Moreover, they also had issues completely reconstructing the thin rings on top of the stool. TSR (Aroudj et al., 2017) was able to fit to the available points but is unable to complete structures such as bottom surface of the stool and it's cylindrical legs, where no observations exist. This shows how our method utilizes local shape priors to complete partially scanned shapes.



**(a)** Input Points

**(b)** MPU

**(c)** TSDF Fusion

**(d)** SSD

**(e)** PSR

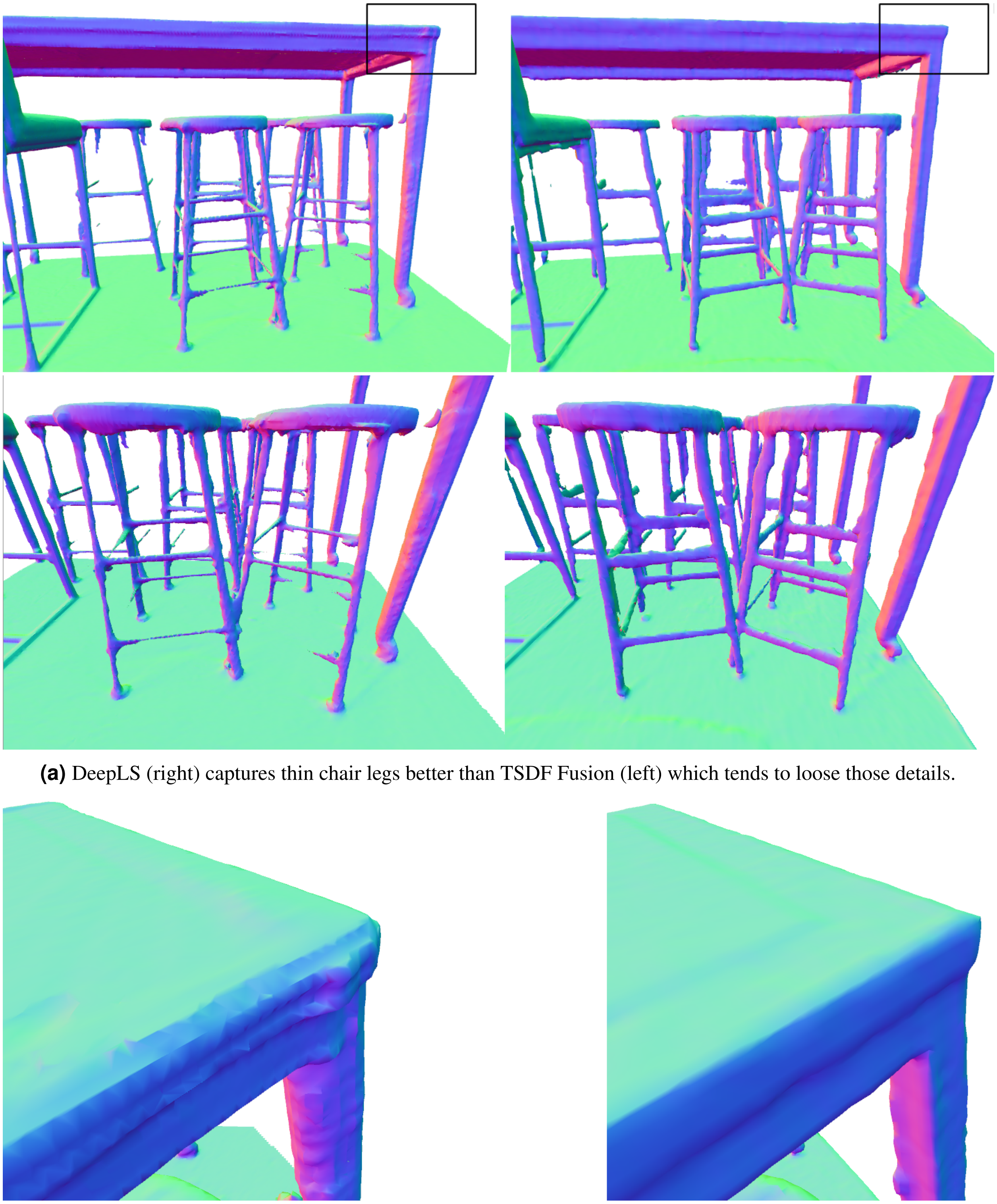
**(f)** PFS

**(g)** TSR

**(h) DeepLS (Ours)**

**(i)** Close-up view. From left to right: RGB, Input Points, TSR and DeepLS reconstructions.

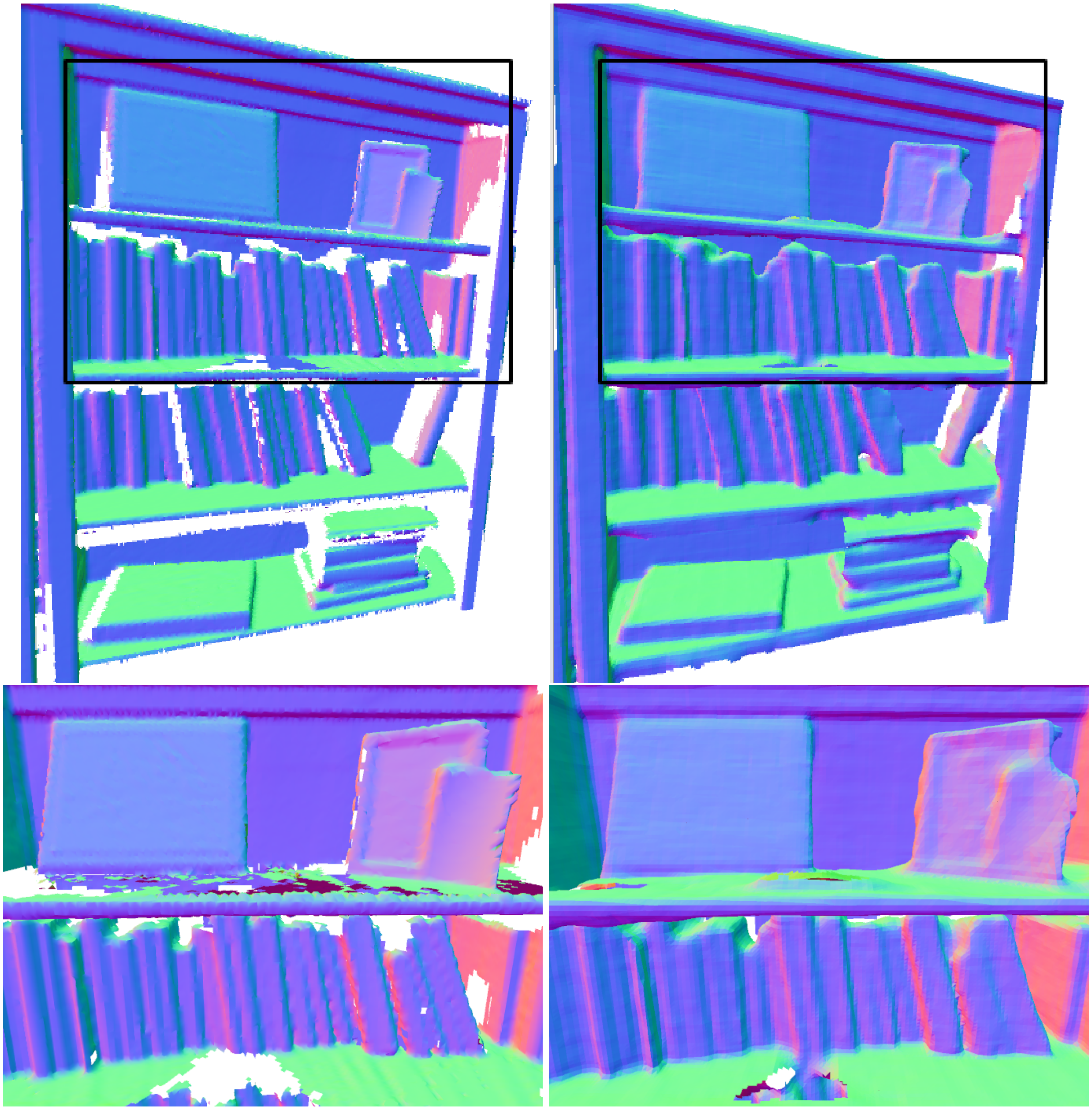
Figure 5.16: We show qualitative comparison of DeepLS against other 3D Reconstruction techniques on a thin and incomplete dataset. We included MPU (Ohtake et al., 2005), KinectFusion (Newcombe et al., 2011a), SSD Calakli and Taubin (2011), PSR Kazhdan and Hoppe (2013), PFS Ummenhofer and Brox (2015) and TSR Aroudj et al. (2017) methods in this comparison. Notice, how most of the methods fail to build thin surfaces in this dataset. Although, TSR fits to the thin parts but is unable to complete structures such as the back and cylindrical legs of the stool (It fits planes to represent cylindrical legs,Fig. 5.16i). Whereas, in comparison DeepLS reconstructs thin structures and also completes them.



**(a)** DeepLS (right) captures thin chair legs better than TSDF Fusion (left) which tends to loose those details.

**(b)** Zoomed view of region marked with black box in (a). DeepLS (right) represents sharper corners and smoother planes than TSDF Fusion (left).

Figure 5.17: Qualitative comparison of TSDF Fusion (left) with DeepLS (right) on real scanned data prepared using the structured light sensor system discussed in (Straub et al., 2019a). The figure (b) is the magnified region marked with black box in figure (a).



**(a)** TSDF Fusion (Newcombe et al., 2011a)

**(b)** DeepLS

Figure 5.18: We show the scene reconstruction quality of DeepLS vs TSDF Fusion (Newcombe et al., 2011a) on a partially scanned real scene dataset using the depth system described in Replica Dataset(Straub et al., 2019a). This figure shows that DeepLS provides better local shape completion than TSDF Fusion. The bottom row represents the zoomed in view marked with black box in the top row.

We show additional qualitative results on real scanned data in Fig. 5.17 and Fig. 5.18. Both scenes showed in the figures were captured using a handheld structured light sensor system described for the stool scene. It can be seen that DeepLS succeeds in representing small details like the bars of chairs while TSDF Fusion tends to loose these details. Also, we observe sharper corners (c.f. 5.17b) and more complete surfaces (c.f. 5.18b) with DeepLS.

### 5.8 Preliminary work on Object-centric Representations

We conducted a preliminary experiment, connecting our local shape codes to a global shape code to encode whole objects. Given the object-centric latent code as $\mathbf{z}_g$ and an additional 6-layer MLP $f_\theta$, we encode a field of local codes $\mathbf{z}_i$ by minimizing an additional loss

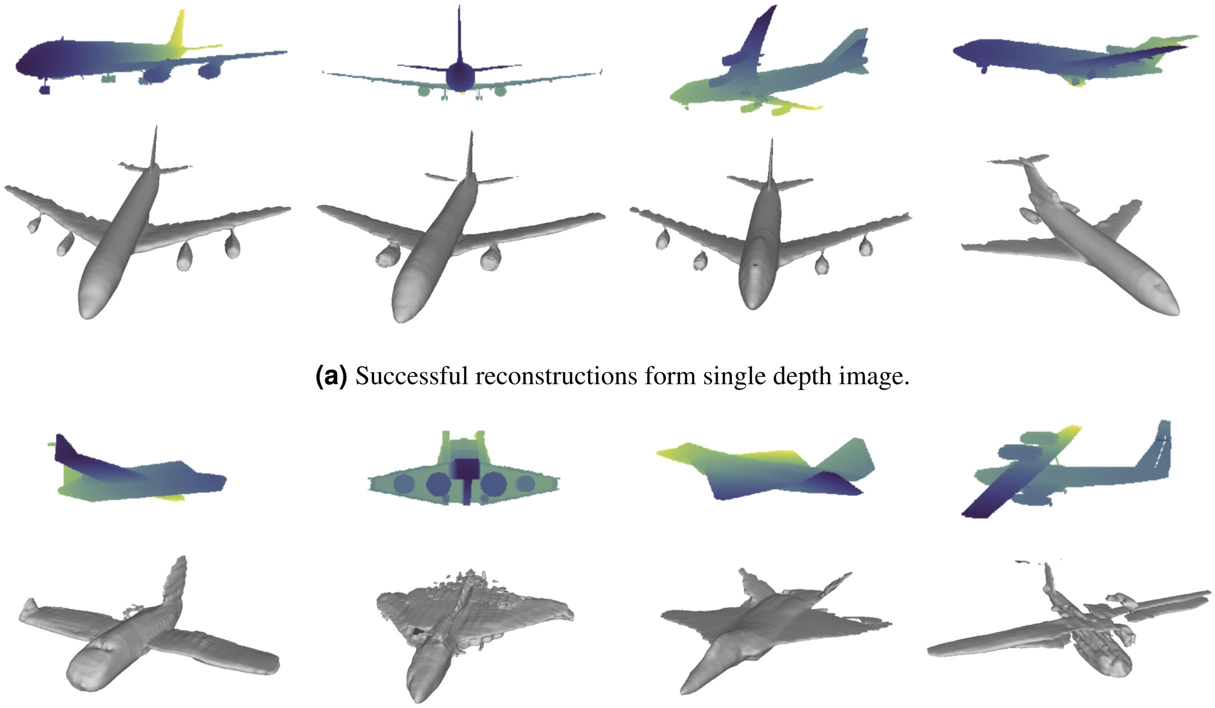
$$\mathcal{L}_g = ||\mathbf{z}_i - f_\theta(\mathbf{c}(i), \mathbf{z}_g)||_1, \tag{5.6}$$

where $\mathbf{c}(i) \in \mathbb{R}^3$ is the coordinate of voxel block $i$. We use the usual training and inference procedures, that is, optimizing code $\mathbf{z}_g$ and $\theta$ during training stage and only $\mathbf{z}_g$ during inference. Training local and global encoders and codes jointly showed better convergence than training them in two independent steps. Doing that, local and global MLPs can agree on a matching intermediate representation of local codes during training.

To decode representations that were obtained by encoding a partial observation, we perform marching cubes on a combination of local codes $\mathbf{z}_i$ and codes $f_\theta(\mathbf{c}(i), \mathbf{z}_g)$ obtained from the global code: For each block $i$ for which observations exist, we use the obtained local codes $\mathbf{z}_i$, for all others we fall back to $f_\theta(\mathbf{c}(i), \mathbf{z}_g)$.

The completion experiment follows the description from Park *et al*. Park et al. (2019). The model was trained on the ShapeNetV2 planes training set and obtained depth scans from a random view point for every element of the ShapeNetV2 planes test set. To encode the partial observation, we compute three samples for each depth value: The point obtained from the depth sample with zero SDF and small positive/negative offsets of those along local surface normals. The normals are naively computed from the depth image.

Results, including failure cases, are shown in Fig. 5.19. The results confirm that deep architectures can be built upon DeepLS codes and that global object patterns can be learned from the



**(a)** Successful reconstructions form single depth image.

**(b)** Failure cases.

Figure 5.19: Completion from single-view depth images on ShapeNetV2 planes. In both figures, the top row shows the depth image and the bottom row the reconstructed and completed model from a different perspective. Figure (b) shows failure cases of individual test examples that differ from the training distribution.

codes, as can be seen by the symmetric reconstructions of plane turbines. However, we also observed that for examples which are slightly out of training distribution, the MLP fails to generate matching local codes, since consistency between neighboring blocks is not built into the architecture. The failure cases in Fig. 5.19b show those examples. Also the information about local details coming from the global code is not as accurate as from the directly fitted DeepLS codes yet. We leave further investigation and improvement of such extended architectures as future work.

### 5.9 Conclusion

In this work we presented DeepLS, a method to combine the benefits of volumetric fusion and deep shape priors for 3D surface reconstruction from depth observations. Key to the success of this approach is the decomposition of large surfaces into local shapes. This decomposition allowed us to reconstruct surfaces with higher accuracy and finer detail than traditional SDF fusion techniques, while simultaneously completing unobserved surfaces, all using less memory than storing the full SDF volume would require. Compared to recent object-centric shape learning approaches, our local shape decomposition leads to greater efficiency for both training and inference while improving surface reconstruction accuracy by an order of magnitude.

# CHAPTER 6: DISCUSSION

In this dissertation, we explored various ways in which priors learned using artificial neural networks can improve state-of-the-art in 3D reconstruction and depth estimation. In this chapter, we conclude this dissertation and briefly discuss the known limitations of our proposed work and possible future directions to further improve state-of-the-art in 3D reconstruction.

## 6.1 Conclusion

Estimating depth information in the form of range or depth images from the environment around us is an important part of the 3D reconstruction system. In this dissertation (Chapter 1), we outline the challenges and the limitations of the existing methods in providing effective and ubiquitous solutions to depth estimation tasks. In Chapter 3, we proposed *StereoDRNet* (Chabra et al., 2019), that shows how using deep learning priors can help us in obtaining state-of-the-art depth estimation in both outdoor and indoor scenes from a single passive stereo pair. We further demonstrate that our passive stereo system, when used for building 3D scene reconstructions in challenging indoor scenes with homogeneous texture, surfaces with shadows and specular reflections, approaches the quality of state-of-the-art structured light systems (Whelan et al., 2018). This motivates the use of passive cameras for indoor scene reconstruction as they potentially provide small, cheap, and low power requirement alternative to the use of active depth sensors which are relatively bigger, expensive and have high power requirement.

Technically, *StereoDRNet* is a novel network architecture that uses global spatial pooling and dilated residual cost filtering techniques to approximate the underlying geometry even in challenging scenarios. Furthermore, the proposed refinement network produces geometrically consistent disparity maps with the help of occlusion and view consistency cues. The use of perfect synthetic data and careful filtering of real training data enables the system to recover thin structures and sharp object boundaries.

The second crucial part of this dissertation is to improve state-of-the-art in the context of 3D surface representation. In Chapter 1, we briefly discuss the limitations of the methods that try to provide global and local *implicit surface* representation. Although most of the traditional global methods promise to provide complete, continuous, and consistent 3D surface representation but they are prone to errors and artifacts in the presence of noise, outliers, partial observations, and thin features in the input data. Moreover, as these systems tend to solve a large, computationally expensive, global solution, so they often are limited by available hardware resources and are hard to extend to very large scenes. While the local method has advantages of being more robust to noise and can easily extend to large scenes but they do not hold shape generation and completion properties as they usually approximate *implicit surface* with a scalar in very small grid cells. Moreover, such methods have difficulty representing and resolving thin surfaces.

In Chapter 5, we extend the prior work on learned deep generative modeling, DeepSDF (Park et al., 2019), to efficiently work on large scenes. DeepSDF can learn compressed *implicit surface* representation of classes of 3D shapes and is very robust to noise, outliers, and incomplete input data. Although, much as it's classical counterparts, DeepSDF is only shown to effectively learn simple shapes such as sofa, chairs, lamps, etc. In our experiments in Chapter 5, we found that DeepSDF takes several days to learn little complex shapes such as a Stanford Bunny. Thus, such methods might take an impractical amount of time and parameters to extend to large scenes. To tackle this problem in this thesis, we propose *DeepLS*, that learn priors on local shapes in the regular 3D grids and enforce the local neighborhood consistencies on them to achieve globally consistent surfaces. As the space of possible local shapes is much smaller than of general shapes and scenes, the learning process is simple and hits very early convergence with orders of magnitude smaller number of parameters compared to DeepSDF. We illustrate that using deep local shape priors; we can represent the general 3D scene environments both more accurately and more efficiently than the previous state-of-the-art 3D reconstruction methods. Furthermore, we show that *DeepLS*, can produce a complete 3D reconstruction of partially scanned thin objects more effectively than other surface representation methods. The extent of shape generation and completion is limited to the size of the local shapes; in practice, we keep them several times larger than other local methods such as (Curless and Levoy, 1996; Newcombe et al., 2011a).

## 6.2 Limitations and Future Work

In this section, we discuss several limitations of the work that was proposed in this dissertation. For each of these limitations, we propose possible future work directions that might help to tackle these problems.

### 6.2.1 Learning Multi-View Stereo

In our proposed method for reconstructing 3D environments using a passive stereo camera, we just use two views to solve the depth extraction task. Although many stereo views can be incorporated to improve the stereo matching procedure using Multi-View Stereo (MVS) (Goesele et al., 2006) technique. While several learning methods such as MVSNet (Yao et al., 2018) that learn to filter stereo cost volume over several views, have been proposed but in our experiments (in Chapter 4) we found that our method *StereoDRNet* produces better results than MVSNet in 3D reconstruction task. Thus, it is still challenging to learn 2D image features that are robust to match across variable views as compared to fixed views in a rigid stereo pair. We believe that finding solutions to this particular problem is an interesting direction for the future.

In MVS, each stereo matching pair suffers from occlusion. Thus, incorporating techniques to learn occlusion-aware filtering of stereo cost volume is an important task. In the literature, several methods have been proposed to handle occlusion (Newcombe, 2012). Similarly, end to end MVS learning methods need schemes to handle occlusion properly. We handled occlusion in our work *StereoDRNet* for two-view stereo, and we believe that such an approach can be extended for multi-view stereo tasks.

### 6.2.2 Real-time Learning Stereo

In Chapter 3, we demonstrate significant improvement in efficiency over the state-of-the-art in a depth estimation task. Although, still our proposed system is not real-time. Pruning the network size and switching to integer precision seem obvious choices to decrease the computation load and improve the efficiency of the system. We believe that the light-weight refinement network proposed by us in Chapter 3 can be used along with several faster stereo matching techniques such as SGM (Hirschmuller, 2008). This will help these systems to produce better accuracy with only a marginal loss of efficiency.

### 6.2.3 Generalization in Learning Stereo

While in Chapter 3, we show that our stereo system, once trained on a large synthetic dataset can be re-tuned on a typical stereo pair with the help of a small training set (less than 200 views) but still obtaining ground truth depth maps for each stereo rig is a tedious task. Using a variety of stereo rigs with variable *baselines* and *field of views* in our training set, we can make the trained model self-sufficient, as shown in monocular depth estimation (Ummenhofer et al., 2017). Whether it is possible to train such a generalized stereo system with similar or better accuracy compared to the re-tuned system remains an interesting challenge for future work.

### 6.2.4 Estimating Confidence Maps in Learned Stereo

For tasks such as a fusion of several depth maps into a consistent 3D reconstruction model, the estimation of confidence for each depth prediction is an important task. Volumetric integration methods such as KinectFusion (Newcombe et al., 2011a) expose such confidence parameters to correctly weigh the depth measurements from several views. There has been a very insightful analysis of obtaining uncertainty on the optical flow predictions in (Ilg et al., 2018a). We believe that estimating such uncertainty in depth prediction from learned stereo can help in producing effective confidence maps. Such confidence maps can then be used in volumetric fusion algorithms for 3D reconstruction tasks.

### 6.2.5 Hierarchical Learning of Shapes

It is very interesting to see the scope of utilizing the learning of multi-resolution shapes. Traditionally, this idea has been utilized by several 3D Reconstruction algorithms, such as PSR (Kazhdan et al., 2006). The higher resolution implicit function adds high-frequency details to lower resolution implicit function. Octree representation is often used to represent the hierarchy of multi-resolution shape functions; this tree structure can also be utilized to handle multi-scale input data for adaptive allocation of shapes depending on the input scale. Learning to represent a scene at multi-resolution also improves the chances of producing complete reconstruction, especially in the regions with sparse input data.

### 6.2.6 Real-time Optimization of Local Shape Functions

Real-time 3D reconstruction systems have several applications in Augmented and Virtual Reality, Tele-presence, robot navigation, 3D mapping and localization, etc. While, in Chapter 5 we introduced learning of local shape functions that has several advantages over other volumetric integration methods such KinectFusion (Newcombe et al., 2011a) but our proposed system *DeepLS* is not real-time. By iteratively optimizing the local shape functions we can try to make the system real-time. Thus, each depth map can iteratively help us allocate new local shapes and also improve the prediction of existing ones. We now provide detailed formalization of this idea with the hope that it will help in the implementation.

Consider the sample set $S = \{s_0, s_1, s_2, ...s_N\}$ containing $N$ samples for a local shape. If all $N$ samples are available at once for test then our simple but optimal loss function is

$$L_{opt} = \sum_{i=1}^{N} ||\hat{\psi}_i(\hat{C}) - \psi_i||_1 + ||\hat{C}||_2^2 \tag{6.1}$$

where $\hat{\psi}_i$ is the predicted SDF, and $\psi_i$ is the ground truth SDF. $\hat{C}$ is the optimized code.

Now let's consider that we have $k$ subsets of $S$ labeled as $S_j$. The individual optimized codes for each subset $S_j$ are labeled as $C_j$. Let's consider the fused code to be

$$C_{fused} = \frac{\sum_{j=1}^{k} C_j}{k} \tag{6.2}$$

The consistency loss that pushes the fused codes $C_{fused}$ to be close to the optimal one $\hat{C}$ can be formulated as

$$L_{consistency} = ||C_{fused} - \hat{C}||_2^2 \tag{6.3}$$

The overall loss in training that can enable the fusion of codes can be formulated as

$$L_{fused} = \sum_{j}^{k} \sum_{i \in S_j} ||\hat{\psi}_i(C_j) - \psi_i||_1 + \lambda || \frac{\sum_{j=1}^{k} C_j}{k} - \hat{C}||_2^2 \tag{6.4}$$

where $\hat{\psi}_i(C_j)$ is the predicted sdf by the code $C_j$ and $\lambda$ is the consistency weight.

Using Eq. 6.3 we can derive

$$L_{fused} = \sum_{j}^{k} \sum_{i \in S_j} ||\psi_i(C_j) - \psi_i||_1 + \lambda L_{consistency} \tag{6.5}$$

# REFERENCES

(1996). The Stanford 3D Scanning Repository. `http://graphics.stanford.edu/data/3Dscanrep/`.

Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. (2017). Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*.

Aroudj, S., Seemann, P., Langguth, F., Guthe, S., and Goesele, M. (2017). Visibility-consistent thin surface reconstruction using multi-scale kernels. *ACM Transactions on Graphics (TOG)*, 36(6):187.

Baker, H. H. (1982). Depth from edge and intensity based stereo. Technical report, Stanford Univ CA Dept of Computer Science.

Ben-Hamu, H., Maron, H., Kezurer, I., Avineri, G., and Lipman, Y. (2018). Multi-chart generative surface modeling. In *SIGGRAPH Asia 2018 Technical Papers*, page 215. ACM.

Bleyer, M., Rhemann, C., and Rother, C. (2011). Patchmatch stereo-stereo matching with slanted support windows. In *Bmvc*, volume 11, pages 1–11.

Blinn, J. F. (1982). A generalization of algebraic surface drawing. *ACM transactions on graphics (TOG)*, 1(3):235–256.

Bloomenthal, J., Bajaj, C., Blinn, J., Wyvill, B., Cani, M.-P., Rockwood, A., and Wyvill, G. (1997). *Introduction to implicit surfaces*. Morgan Kaufmann.

Buhmann, M. D. (2003). *Radial basis functions: theory and implementations*, volume 12. Cambridge university press.

Calakli, F. and Taubin, G. (2011). Ssd: Smooth signed distance surface reconstruction. In *Computer Graphics Forum*, volume 30, pages 1993–2002. Wiley Online Library.

Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. (2001). Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM.

Chabra, R., Lenssen, J. E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., and Newcombe, R. (2020). Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. *arXiv preprint arXiv:2003.10983*.

Chabra, R., Straub, J., Sweeney, C., Newcombe, R., and Fuchs, H. (2019). Stereodrnet: Dilated residual stereonet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11786–11795.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., and Zhang, Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. (2015). ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Chang, J.-R. and Chen, Y.-S. (2018). Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418.

Chen, H. and Bhanu, B. (2007). 3d free-form object recognition in range images using local surface patches. *Pattern Recognition Letters*, 28(10):1252–1262.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848.

Chen, Z. and Zhang, H. (2019). Learning implicit fields for generative shape modeling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE.

Choy, C. B., Xu, D., Gwak, J., Chen, K., and Savarese, S. (2016). 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer.

Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images.

Dai, A. and Nießner, M. (2019). Scan2mesh: From unstructured range scans to 3d meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5574–5583.

Dai, A., Ruizhongtai Qi, C., and Nießner, M. (2017). Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5868–5877.

Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., and Tagliasacchi, A. (2019). Cvxnets: Learnable convex decomposition.

Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625.

Fabbri, R. and Kimia, B. (2010). 3d curve sketch: Flexible curve-based stereo reconstruction and calibration. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1538–1545. IEEE.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2006). Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54.

Felzenszwalb, P. F. and Zabih, R. (2010). Dynamic programming and graph algorithms in computer vision. *IEEE transactions on pattern analysis and machine intelligence*, 33(4):721–740.

Florence, P. R., Manuelli, L., and Tedrake, R. (2018). Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*.

Fuhrmann, S. and Goesele, M. (2014). Floating scale surface reconstruction. *ACM Transactions on Graphics (ToG)*, 33(4):46.

Gal, R., Shamir, A., Hassner, T., Pauly, M., and Cohen-Or, D. (2007). Surface reconstruction using local shape priors. In *Symposium on Geometry Processing*, number CONF, pages 253–262.

Geiger, A., Lenz, P., and Urtasun, R. (2012). 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.

Geiger, A., Roser, M., and Urtasun, R. (2010). Efficient large-scale stereo matching. In *Asian conference on computer vision*, pages 25–38. Springer.

Genova, K., Cole, F., Sud, A., Sarna, A., and Funkhouser, T. (2019a). Deep structured implicit functions. *arXiv preprint arXiv:1912.06126*.

Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W. T., and Funkhouser, T. (2019b). Learning shape templates with structured implicit functions. *arXiv preprint arXiv:1904.06447*.

Gkioxari, G., Malik, J., and Johnson, J. (2019). Mesh r-cnn. *arXiv preprint arXiv:1906.02739*.

Goesele, M., Curless, B., and Seitz, S. M. (2006). Multi-view stereo revisited. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2402–2409. IEEE.

Groueix, T., Fisher, M., Kim, V. G., Russell, B. C., and Aubry, M. (2018). Atlasnet: A papier-m\^ ach\'e approach to learning 3d surface generation. *arXiv preprint arXiv:1802.05384*.

Handa, A., Whelan, T., McDonald, J., and Davison, A. (2014). A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China.

Häne, C., Tulsiani, S., and Malik, J. (2017). Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier.

Hirschmuller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 71–78.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

Hu, X. and Mordohai, P. (2012). A quantitative evaluation of confidence measures for stereo vision. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2121–2133.

Ilg, E., Cicek, O., Galesso, S., Klein, A., Makansi, O., Hutter, F., and Brox, T. (2018a). Uncertainty estimates and multi-hypotheses networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 652–667.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE conference on computer vision and pattern recognition (CVPR)*, volume 2, page 6.

Ilg, E., Saikia, T., Keuper, M., and Brox, T. (2018b). Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *European Conference on Computer Vision (ECCV)*.

Jancosek, M. and Pajdla, T. (2011). Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128. IEEE.

Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., and Aanæs, H. (2014). Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7.

Kazhdan, M. and Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13.

Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A. (2013). Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 1–8. IEEE.

Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., and Bry, A. (2017). End-to-end learning of geometry and context for deep stereo regression. *CoRR, vol. abs/1703.04309*.

Khamis, S., Fanello, S., Rhemann, C., Kowdle, A., Valentin, J., and Izadi, S. (2018). Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. *arXiv preprint arXiv:1807.08865*.

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

Klaus, A., Sormann, M., and Karner, K. (2006). Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 15–18. IEEE.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10. IEEE Computer Society.

Kolmogorov, V. and Zabih, R. (2001). Computing visual correspondence with occlusions using graph cuts. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 508–515. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Labatut, P., Pons, J.-P., and Keriven, R. (2009). Robust and efficient surface reconstruction from range data. In *Computer graphics forum*, volume 28, pages 2275–2290. Wiley Online Library.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Liang, Z., Feng, Y., Chen, Y., and Zhang, L. (2018). Learning for disparity estimation through feature constancy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2811–2820.

Liao, Y., Donne, S., and Geiger, A. (2018). Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048.

Meltzer, J. and Soatto, S. (2008). Edge descriptors for robust wide-baseline correspondence. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. (2019). Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470.

Michalkiewicz, M., Pontes, J. K., Jack, D., Baktashmotlagh, M., and Eriksson, A. (2019). Deep level sets: Implicit surface representations for 3d shape inference. *arXiv preprint arXiv:1901.06802*.

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163.

Newcombe, R. (2012). *Dense visual SLAM*. PhD thesis, Imperial College London.

Newcombe, R. A. and Davison, A. J. (2010). Live dense reconstruction with a single moving camera. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1498–1505. IEEE.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011a). Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011b). Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE.

Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., and Seidel, H.-P. (2005). Multi-level partition of unity implicits. In *Acm Siggraph 2005 Courses*, pages 173–es.

Pang, J., Sun, W., Ren, J. S., Yang, C., and Yan, Q. (2017). Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *ICCV Workshops*, volume 7.

Papandreou, G., Kokkinos, I., and Savalle, P.-A. (2015). Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 390–399.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. *arXiv preprint arXiv:1901.05103*.

Pfister, H., Zwicker, M., Van Baar, J., and Gross, M. (2000). Surfels: Surface elements as rendering primitives. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 335–342. ACM Press/Addison-Wesley Publishing Co.

Pock, T., Unger, M., Cremers, D., and Bischof, H. (2008). Fast and exact solution of total variation models on the gpu. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE.

Pradeep, V., Rhemann, C., Izadi, S., Zach, C., Bleyer, M., and Bathiche, S. (2013). Monofusion: Real-time 3d reconstruction of small scenes with a single web camera. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 83–88. IEEE.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660.

Rao, D., Chung, S.-J., and Hutchinson, S. (2012). Curveslam: An approach for vision-based navigation without point features. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4198–4204. IEEE.

Ricao Canelhas, D., Schaffernicht, E., Stoyanov, T., Lilienthal, A., and Davison, A. (2017). Compressed voxel-based mapping using unsupervised learning. *Robotics*, 6(3):15.

Riegler, G., Osman Ulusoy, A., and Geiger, A. (2017). Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268.

Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., and Li, H. (2019). Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *arXiv preprint arXiv:1905.05172*.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42.

Scharstein, D. and Szeliski, R. (2003). High-accuracy stereo depth maps using structured light. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE.

Schmidt, T., Newcombe, R., and Fox, D. (2016). Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427.

Schönberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schönberger, J. L., Zheng, E., Pollefeys, M., and Frahm, J.-M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*.

Schöps, T., Schönberger, J. L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., and Geiger, A. (2017). A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schops, T., Schonberger, J. L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., and Geiger, A. (2017). A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3260–3269.

Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528. IEEE.

Sinha, A., Bai, J., and Ramani, K. (2016). Deep learning 3d shape surfaces using geometry images. In *European Conference on Computer Vision*, pages 223–240. Springer.

Song, X., Zhao, X., Hu, H., and Fang, L. (2018). Edgestereo: A context integrated residual pyramid network for stereo matching. *arXiv preprint arXiv:1803.05196*.

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H. M., Nardi, R. D., Goesele, M., Lovegrove, S., and Newcombe, R. (2019a). The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*.

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., et al. (2019b). The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*.

Stühmer, J., Gumhold, S., and Cremers, D. (2010). Real-time dense geometry from a handheld camera. In *Joint Pattern Recognition Symposium*, pages 11–20. Springer.

Stutz, D. and Geiger, A. (2018). Learning 3d shape completion from laser scan data with weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1955–1964.

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2018). Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943.

Sun, J., Ovsjanikov, M., and Guibas, L. (2009). A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library.

Sun, S., Cao, Z., Zhu, H., and Zhao, J. (2019). A survey of optimization methods from a machine learning perspective. *IEEE transactions on cybernetics*.

Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2017). Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096.

Tulyakov, S., Ivanov, A., and Fleuret, F. (2018). Practical deep stereo (pds): Toward applications-friendly deep stereo matching. *arXiv preprint arXiv:1806.01677*.

Ummenhofer, B. and Brox, T. (2013). Point-based 3d reconstruction of thin objects. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 969–976.

Ummenhofer, B. and Brox, T. (2015). Global, dense multiscale reconstruction for a billion points. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1341–1349.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2017). Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on computer vision and pattern recognition (CVPR)*, volume 5, page 6.

Wei, W. and Ngan, K. N. (2005). Disparity estimation with edge-based matching and interpolation. In *2005 International Symposium on Intelligent Signal Processing and Communication Systems*, pages 153–156. IEEE.

Whelan, T., Goesele, M., Lovegrove, S. J., Straub, J., Green, S., Szeliski, R., Butterfield, S., Verma, S., and Newcombe, R. (2018). Reconstructing scenes with mirror and glass surfaces. *ACM Transactions on Graphics (TOG)*, 37(4):102.

Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., and Davison, A. (2015). Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems.

Williams, F., Schneider, T., Silva, C., Zorin, D., Bruna, J., and Panozzo, D. (2019). Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10130–10139.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.

Xie, C.-W., Zhou, H.-Y., and Wu, J. (2018). Vortex pooling: Improving context representation in semantic segmentation. *arXiv preprint arXiv:1804.06242*.

Xu, Q., Wang, W., Ceylan, D., Mech, R., and Neumann, U. (2019). Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *arXiv preprint arXiv:1905.10711*.

Yang, G., Zhao, H., Shi, J., Deng, Z., and Jia, J. (2018). Segstereo: Exploiting semantic information for disparity estimation. *arXiv preprint arXiv:1807.11699*.

Yang, Y., Feng, C., Shen, Y., and Tian, D. (2017). Foldingnet: Interpretable unsupervised learning on 3d point clouds. *arXiv preprint arXiv:1712.07262*.

Yao, Y., Luo, Z., Li, S., Fang, T., and Quan, L. (2018). Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783.

Yu, F., Koltun, V., and Funkhouser, T. A. (2017). Dilated residual networks. In *CVPR*, volume 2, page 3.

Yuan, W., Khot, T., Held, D., Mertz, C., and Hebert, M. (2018). Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE.

Zach, C., Pock, T., and Bischof, H. (2007). A globally optimal algorithm for robust tv-l 1 range image integration. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Zbontar, J., LeCun, Y., et al. (2016). Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2.

Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., and Funkhouser, T. (2017). 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1802–1811.

Zhang, C., Li, Z., Cheng, Y., Cai, R., Chao, H., and Rui, Y. (2015). Meshstereo: A global stereo model with mesh alignment regularization for view interpolation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2057–2065.

Zhang, Y., Khamis, S., Rhemann, C., Valentin, J., Kowdle, A., Tankovich, V., Schoenberg, M., Izadi, S., Funkhouser, T., and Fanello, S. (2018). Activestereonet: end-to-end self-supervised learning for active stereo systems. *arXiv preprint arXiv:1807.06009*.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334.

Zhong, Y. (2009). A shape descriptor for 3d object recognition. In *Proceedings ICCV 2009 Workshop 3DRR*, volume 6.

Zhou, Q.-Y. and Koltun, V. (2013). Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (ToG)*, 32(4):1–8.

Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R., and Kolb, A. (2018). State of the art on 3d reconstruction with rgb-d cameras. In *Computer graphics forum*, volume 37, pages 625–652. Wiley Online Library.