Indirect 3D Reconstruction Through Appearance Prediction

Clément Godard

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Engineering

of

University College London.

Department of Computer Science

University College London

September 20, 2018

I, Clément Godard, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

As humans, we easily perceive shape and depth, which helps us navigate our environment and interact with objects around us. Automating these abilities for computers is critical for many applications such as self-driving cars, augmented reality or architectural surveying.

While active 3D reconstruction methods, such as laser scanning or structured light can produce very accurate results, they are typically expensive and their use cases can be limited. In contrast, passive methods that make use of only easily captured photographs, are typically less accurate as mapping from 2D images to 3D is an under-constrained problem. In this thesis we will focus on passive reconstruction techniques.

We explore ways to get 3D shape from images in two challenging situations: 1) where a collection of images features a highly specular surface whose appearance changes drastically between the images, and 2) where only one input image is available. For both cases, we pose the reconstruction task as an indirect problem.

In the first situation, the rapid change in appearance of highly specular objects makes it infeasible to directly establish correspondences between images. Instead, we develop an indirect approach using a panoramic image of the environment to simulate reflections, and recover the surface which best predicts the appearance of the object.

In the second situation, the ambiguity inherent in single-view reconstruction is typically solved with machine learning, but acquiring depth data for training is both difficult and expensive. We present an indirect approach, where we train a neural network to regress depth by performing the proxy task of predicting the appearance

of the image when the viewpoint changes.

We prove that highly specular objects can be accurately reconstructed in uncontrolled environments, producing results that are 30% more accurate compared to the initialisation surface. For single frame depth estimation, our approach improves object boundaries in the reconstructions and significantly outperforms all previously published methods. In both situations, the proposed methods shrink the accuracy gap between camera-based reconstruction versus what is achievable through active sensors.

Impact Statement

In this thesis we present three novel 3D reconstruction methods. All the work presented here was disseminated via conferences, online release on arXiv as well as open source code releases.

Chapter 3 In this chapter we show a method to reconstruct mirror-like surfaces in uncontrolled environments, using photographs without the need for additional hardware unlike most previous methods. The method significantly simplifies and lowers the cost of capturing specular objects and has potential applications in art preservation, archeology and the entertainment industry. This work was presented as an oral at the 3DV conference in 2015.

Chapter 4 In this chapter we use stereo pairs to train a neural network to predict depth from single images. This work was presented as an oral at the CVPR conference in 2017. The method was state of the art when released and demonstrated that it was possible to reach a high accuracy in depth estimation using only self-supervision, meaning in this case that depth was not used when training. This opens the door to a lot of applications where acquiring ground truth depth is difficult and expensive such as with self-driving cars, robotics, or entertainment. The code was made open source on GitHub, proved to be quite popular and significantly helped the visibility of the method. It also made it easy for others to build upon it and a number of methods based on the released code were presented at other conferences in 2018. A patent was applied for in 2017 and acquired by Niantic in 2018.

Chapter 5 In this chapter we extend the method from Chapter 4, significantly improving its accuracy as well as allowing it to train using image sequences only. A patent has been applied for by Niantic and the code will be released.

Publications

The work in this report appears in the following publications:

- Clément Godard, Peter Hedman, Wenbin Li, and Gabriel J Brostow. Multi-view reconstruction of highly specular surfaces in uncontrolled environments. In the IEEE International Conference on 3D Vision (3DV), 2015. IEEE, 2015.

- Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

- Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Digging Into Self-Supervised Monocular Depth Estimation. arXiv preprint arXiv:1806.01260, 2018.

Acknowledgements

First and foremost I would like to thank my advisor, Gabriel Brostow for his support during my PhD. His enthusiasm and drive allowed me to work on exciting topics. I am very grateful to have worked with Oisín Mac Aodha, who made sure that my work made it into publications. Thank you to the prism group members at UCL for all their help and advice. I would also like to thank my PhD examiners for their time and comments.

I am thankful to both David Gallup and Matt Uyttendaele for allowing me to spend two fantastic summers in Seattle while interning at Google and Facebook.

I would like to thank my friends and family. My parents and my sister for being present for me throughout the years. To Tania for being here during the not-so-fun first few months. To Clément and Tracy for their invaluable friendship and for all the French food. Sara for all the music. Moos for all the good talks. To Corny, Tara and Lucy for being great friends. To Peter for making sure I didn't sleep enough and for making me believe in myself. To everyone else that I forgot to include. And finally to Carolina for everything.

Contents

1 Introduction **10**

2 Previous work **15**

2.1 Surface reconstruction 15

2.1.1 Specular surface reconstruction 15

2.1.2 Strong diffuse components 16

2.2 Monocular depth estimation 17

2.2.1 Learning-Based Stereo 18

2.2.2 Supervised Single Image Depth Estimation 18

2.2.3 Self-Supervised Depth Estimation 19

3 Multi-view Reconstruction of Highly Specular Surfaces in Uncontrolled Environments **21**

3.1 Method 23

3.1.1 Estimating normal distributions 25

3.1.2 Optimization 29

3.2 Implementation and results 33

3.2.1 Synthetic results 34

3.2.2 Real-world results 35

3.3 Conclusion 36

4 Self-Supervised Monocular Depth Estimation with Left-Right Consistency **41**

4.1 Method 43

4.1.1 Depth Estimation as Image Reconstruction 43
4.1.2 Depth Estimation Network 45
4.1.3 Training Loss 46
4.2 Implementation and Results 49
4.2.1 Implementation Details 50
4.2.2 KITTI 51
4.2.3 Stereo 56
4.2.4 Make3D 56
4.2.5 Generalizing to Other Datasets 57
4.2.6 Limitations 57
4.3 Conclusion 58

5 Digging into Self-Supervised Monocular Depth Estimation 60
5.1 Method 64
5.1.1 Self-supervised Training 64
5.1.2 Improved Monocular Depth Estimation 65
5.2 Implementation and Results 69
5.2.1 Implementation Details 69
5.2.2 KITTI 70
5.2.3 Results 71
5.2.4 Additional Results 74
5.2.5 Convergence 75
5.2.6 Effect of moving cars 75
5.2.7 Single scale test time evaluation 76
5.2.8 Odometry 77
5.2.9 Discussion 77
5.3 Conclusion 78

6 Conclusion 82

Bibliography 85

Chapter 1

Introduction

The aim of this thesis is to design algorithms for computers to be able to perceive and reconstruct the 3D world around us. Our goal is to improve the accuracy of such methods when only using photographs as input.

Motivation

The visual reproduction of our world is almost as old as modern humanity itself, with the oldest drawings ever found dating back to more than 17000 years ago. This practice has only grown stronger over time with the invention of painting and sculpture. There seem to be an insatiable need to depict and represent what we observe every day and visual arts are now more popular than ever. In parallel, passing on knowledge to younger generations slowly gave birth to modern science as an organised way to compile our understanding of the universe. As one might expect, these two fields interacted from early on, for instance with the use of oblique perspective in drawings in China from as early as the first century BC [1]. Similarly, the development of the field of optics gave us a better understanding of how light interacts with the environment and how images are formed in our retina. The need to capture and represent our world as accurately as possible in art probably peaked with the Realism movement of the mid-19th century. This period also corresponds to the invention of photography, which very quickly gave the ability to everyone to instantly capture what was in front of them. While photography developed into the ubiquitous medium which we all know today, physical 3D representation stayed in the realm of arts with sculpture. How can we make capture in 3D as easy as

taking a photograph? The need for 3D capture is crucial in applications such as robotics where the interaction with the physical world is essential. To navigate an environment a robot needs to understand how far the walls of a room are, for instance, or how big is a certain object if it needs to grab it. 3D reconstruction also has a growing influence in content creation, such as for the entertainment industry in video games, movies and virtual reality. Artists can now speed up the recreation of a real-world scene or object, typically a manually intensive task.

The development of active methods, such as laser scanning, was an important breakthrough for 3D reconstruction. These active methods come in two main flavours: time-of-flight and triangulation. They both rely on the same principle, which is to project light onto the scene. The first one is targeted at large scenes and fires laser pulses and the device derives the distance from measuring how much time it takes to bounce back. The second variant focuses on closer surfaces and projects a laser dot onto the surface which is then imaged by a camera. The position of the point in the image is then used to triangulate the corresponding 3D point. The triangulation technique can be sped up using a stripe of light or even a full pattern, in which case we talk about structured light [2]. Capturing an entire object or scene object however still takes time. The reliability and accuracy of active light sensors make them a prime choice when capturing so called *ground truth* data. Indeed, outside of the synthetic data case, there is usually no way to capture the *real* underlying surface or depth of a scene.

Another approach to 3D reconstruction is passive reconstruction, which was pioneered in the early 1960s. Researchers such as the neuroscientist Julesz Béla, started to study depth perception in humans from binocular vision [3], and predicted the advent of the field of stereo vision in computer science which soon became a pillar of computer vision. Obtaining depth from a pair of images is essentially a matching problem: how far did each object move between the two images in a stereo pair? Because of its intrinsic ambiguity passive 3D reconstruction is a difficult problem. For example, different parts of the same image can appear identical due to textureless regions or repeated structures. The field is constantly evolving [4, 5] and

modern methods [6, 7] have made multi-view reconstruction into a viable option for reconstruction, with even commercial software appearing over the last decade allowing photogrammetry to make its way into video games and movies.

Goal

This thesis focuses on passive methods because of their ease of access and low cost: most people now own smartphones with cameras which have a high dynamic range and resolution, and keep progressing at a fast pace. Most of these methods rely on comparing pixels or patches *directly* between the input images, and in case of a good match to establish *correspondences*. Such correspondences, usually after a smoothing step, can then be triangulated to recover the depth of a scene.

We deliberately target challenging scenarios for passive reconstruction methods, where the traditional approach of establishing correspondence between images does not apply. We aim to do so by modelling the changes in appearance of the scene or the object when the observing camera moves, which are directly related to the depth and orientation of surfaces.

We hypothesise that passive reconstruction through appearance prediction is viable in situations where direct image correspondence is ill-defined.

Structure

We will show in this thesis, with three different projects, that by a careful consideration of how objects will appear from multiple different viewpoints, we can make passive 3D reconstruction feasible in new domains where traditional direct image matching cannot be used.

In Chapter 3 we focus on the reconstruction of highly specular surfaces from multiple images. Traditional passive methods relying on establishing direct correspondences between images cannot be used here: a matching location between two images does not correspond to a point on the surface but rather a reflection of the environment. We capture the environment with a panoramic image, allowing us to explicitly model reflections and reconstruct the surface which best predicts the appearance of the input images. Our novel method is the first to enable passive

reconstruction of specular surfaces in uncontrolled environments.

In Chapter 4 we tackle reconstruction from a single input image. With only one image available, it is not possible to establish correspondences. As a result, passive reconstruction methods have to rely solely on priors, most of them found through machine learning from large amounts of depth data. While *ground truth* depth data is typically captured using active devices, such as laser scanners, it is difficult to obtain for a large variety of scenes. On the other hand, stereo capture is much more accessible. We thus make use of a large number of stereo pairs, and train our neural network to indirectly infer depth by appearance prediction. Our novel formulation is the first one to use a fully differentiable depth-based image formation model, where one image in the stereo pair is used to predict the other one. Our model is able to achieve not only significantly sharper depth-maps, but is also more accurate than both baseline methods and competing algorithms, even outperforming supervised methods that were trained with ground truth depth data.

Finally in Chapter 5 we further extend the work of Chapter 4 by training with frames from a monocular video, or temporal sequences, instead of stereo pairs, thus reducing the amount of supervision. This problem is more challenging as the network also needs to solve for the relative poses between the source images in order to reconstruct the target one. We however show that by careful considerations and modifications on the appearance prediction model and reconstruction loss, we can achieve more accurate depths and outperform all previously published self-supervised methods.

Context

The work presented in this thesis sits in the larger space of 3D reconstruction. While the method presented in Chapter 3 and the methods presented in Chapters 4 and 5 share little overlap in terms of application domain and techniques used, they all lie within the space of image-based 3D reconstruction methods.

In Chapter 3 we tackle the reconstruction of highly specular surfaces from photographs. This is a particularly difficult task as the large majority of multi-view 3D reconstruction techniques are based on the assumption of photoconsistency. It

assumes that the appearance of a specific surface point will not change when seen from different viewpoints. The complete opposite is true for highly reflective surfaces where such consistency assumptions cannot be made. Thus traditionnal multi-view stereo pipelines will be incapable of reconstructing such surfaces. We thus wondered if it was still possible, only using photographs, to reconstruct such surfaces. We started by looking at how we ourselves tried to understand the shape of specular objects. We made one key observation: we humans are capable of interpretting the shape of such objects and this is partly because we always happen to observe them in context, in an environment which they reflect and that we observe as well. This gives an observer strong cues on the surface shape. However because such observations can still be ambiguous we observed that humans tended to also change their viewpoint. These empirical observations served as the base of the reasoning for the method developped in Chapter 3.

In Chapter 4 and 5 we focus on depth prediction from single images. Because of the nature of the problem it would be quite difficult to derive hand-made rules to be able to predict depth for a large variety of images. Machine learning is thus a very suitable tool for such a task. As described earlier, most previous methods rely on the use of depth data to train models, which is difficult to obtain. Upon seeing the potential of deep models in structured prediction problems the project in Chapter 4 was started as a novel view prediction problem. The premise was the following: when shown a few images of a scene, could a network predict a new view? The challenge resided in how to structure the input data in an interpretable manner. We then realised stereo data was perfect for the related problem of novel view prediction with a single image as input, especially when trained with a single baseline. Finally the key aspect came from asking ouselves: how can we make the output of this network interpretable, or in other words how can we obtain depth? The answer was to use a differentiable sampler which made the base of the work in Chapter 4 and 5.

Chapter 2

Previous work

In this chapter we cover the literature relevant to the two main scenarios this thesis will focus: specular surface reconstruction and single-frame depth estimation.

2.1 Surface reconstruction

Work presented in Chapter 3 fits within the larger problem of specular surface reconstruction from images and is therefore related to a number of techniques such as specular flow, shape from distortion and controlled lighting. Refer to Ihrke *et al.* [8] for a general survey.

2.1.1 Specular surface reconstruction

Specular Flow. In specular flow, surfaces are derived by tracking reflections (or virtual features [9]) of changing illumination on them. Roth and Black [10] apply diffuse flow to reconstruct a reflective surface containing diffuse markings. Adato *et al.* [11] achieve simple surface reconstruction on real-world data using a specular flow formulation. More recently, Sankaranarayanan *et al.* [12] successfully detect parabolic points using image invariants on reflective surfaces. To refine the specular flow approaches, Canas *et al.* [13] propose a linear formulation to simplify the optimisation. They also attempt to recover the surface using a single image [14, 15] and further reduce the number of flows necessary for certain shapes [16].

These methods do not require prior knowledge of the environment but are sensitive to textureless environments (as they rely on specular flow computation), they assume a smooth surface without occlusions and do not handle inter-reflections.

Furthermore, the capture of real-world objects requires a complicated setup where both the camera and the surface are attached to a structure that rotates by a known motion to generate the needed flow. We aim to produce a method which works on arbitrary closed surfaces, does not rely on any custom built hardware and is easily reproducible.

Shape From Distortion. Shape from distortion relies on the deformation of a given pattern reflected by the surface of the object. Bonfort and Sturm [17] use the reflections of a customized pattern to perform voxel carving of the reflective surface. Savarese *et al.* [18, 19] recover a sparse point set and their normals using correspondences between a known checkerboard and its reflection. They also study how humans perceive specular reflections on different surfaces [20]. Tappen [21] studies the recovery of surface normals for smooth, reflective heightfield surfaces from a single photograph in an unknown environment. Balzer *et al.* [22] reconstruct specular surfaces using the reflections of a known pattern displayed at the end of a controllable robot arm. Similarly Weinmann *et al.* [23] successfully reconstruct mirror-like objects using patterns reflected from screens surrounding them and captured with an arc of eleven cameras. Such patterned illumination is also applied to detect defects on glossy materials [24] or used for single shot surface reconstruction [25]. Similarly Tarini *et al.* [26] use multiple colored patterns, achieving high resolution and accurate reconstruction, despite using only one viewpoint. Jacquet *et al.* [27] use the deformation of straight lines reflected in the windows of buildings to recover their near planar normal maps.

Although some of these approaches achieve good reconstruction accuracy, they only apply for calibrated scenes where the illumination is carefully controlled and hundreds of images are typically required, falling under the category of active methods.

2.1.2 Strong diffuse components

The shape of a surface with a specular material can be recovered with controlled illumination if it has a strong diffuse component. Nehab *et al.* [28] mix a dense stereo framework with an analysis of specular consistencies under a structured illumina-

tion to recover surface normals and depth. Tunwattanapong *et al.* [29] recover the shape and normals in a controlled light environment. To deal with non-Lambertian materials and shape reconstruction Schultz [30] uses specular highlights to recover the surface viewed from different angles. That approach has been extended to specular surfaces such as [31], glossy surfaces in [32], and even materials with inter-reflections in [33]. Further, Hernández *et al.* [34] reconstruct non-Lambertian 3D objects using a multi-view photometric stereo framework. Zhou *et al.* [35] extend the idea to support spatially varying isotropic materials.

Although related in spirit to our method as they handle specularities, these methods still assume photoconsistency. Our focus is on perfectly specular surfaces, for which this assumption does not hold.

Shape from Natural Illumination. Johnson and Adelson [36] present a method to recover surface normals of single-color lambertian objects from a single photograph and a model of the environment in the form of a diffuse calibration sphere. Oxholm and Nishino [37] extend this method by using an environment map to fit a probabilistic model to the reflectance of the object [38]. They further expand this work to produce reconstructions from multiple calibrated images [39], which shares many similarities with our work. Most notably, they also represent the incident light field as an environment map, form probability distributions of the surface normals and iteratively refine both the surface geometry and the normals to reconstruct the object. Their method is applicable to a wide range of objects, provided that their materials either contain a strong diffuse component or rough specularities.

However, their focus is not on recovering the shape of objects exhibiting sharp specular reflections. This leaves us without a viable solution for reconstructing such objects in uncontrolled environments.

2.2 Monocular depth estimation

There is a large body of work that focuses on depth estimation from images, either using pairs [40], several overlapping images captured from different viewpoints [41], temporal sequences [42], or assuming a fixed camera, static scene, and chang-

ing lighting [43, 44]. These approaches are typically only applicable when there is more than one input image available of the scene of interest. Here we focus on works related to monocular depth estimation, where there is only a single input image, and no assumptions about the scene geometry or types of objects present are made.

2.2.1 Learning-Based Stereo

The vast majority of stereo estimation algorithms have a data term which computes the similarity between each pixel in the first image and every other pixel in the second image. Typically the stereo pair is rectified and thus the problem of disparity (*i.e.* scaled inverse depth) estimation can be posed as a 1D search problem for each pixel. Recently, it has been shown that instead of using hand defined similarity measures, treating the matching as a supervised learning problem and training a function to predict the correspondences produces far superior results [45, 46]. It has also been shown that posing this binocular correspondence search as a multi-class classification problem has advantages, both in terms of quality of results and speed [47]. Instead of just learning the matching function, Mayer et al. [48] introduced a fully convolutional [49] deep network called DispNet that directly computes the correspondence field between two images. At training time, they attempt to directly predict the disparity for each pixel by minimizing a regression training loss. DispNet has a similar architecture to their previous end-to-end deep optical flow network [50].

The above methods rely on having large amounts of accurate ground truth disparity data and stereo image pairs at training time. This type of data can be difficult to obtain for real world scenes, so these approaches typically use synthetic data for training. Synthetic data is becoming more realistic, *e.g.* [51], but still requires the manual creation of new content for every new application scenario.

2.2.2 Supervised Single Image Depth Estimation

Single-view, or monocular, depth estimation refers to the problem setup where only a single image is available at test time. Saxena et al. [52] proposed a patch-based

model known as Make3D that first over-segments the input image into patches and then estimates the 3D location and orientation of local planes to explain each patch. The predictions of the plane parameters are made using a linear model trained offline on a dataset of laser scans, and the predictions are then combined together using an MRF. The disadvantage of this method, and other planar based approximations, *e.g.* [53], is that they can have difficulty modeling thin structures and, as predictions are made locally, lack the global context required to generate realistic outputs. Instead of hand-tuning the unary and pairwise terms, Liu et al. [54] use a convolutional neural network (CNN) to learn them. In another local approach, Ladicky et al. [55] incorporate semantics into their model to improve dense depth estimation. Karsch et al. [56] attempt to produce more consistent image level predictions by copying whole depth images from a training set. A drawback of this approach is that it requires the entire training set to be available at test time.

Eigen et al. [57, 58] showed that it was possible to produce dense pixel depth estimates using a two scale deep network trained on images and their corresponding depth values. Unlike most other previous work in single image depth estimation, they do not rely on hand crafted features or an initial over-segmentation and instead learn a representation directly from the raw pixel values. Several works have built upon the success of this approach using techniques such as CRFs to improve accuracy [59], changing the loss from regression to classification [60], using other more robust loss functions [61], and incorporating strong scene priors in the case of the related problem of surface normal estimation [62]. Again, like the previous stereo methods, these approaches rely on having high quality, pixel aligned, ground truth depth at training time, which is typically captured using active techniques.

2.2.3 Self-Supervised Depth Estimation

Recently, a small number of deep network based methods for novel view synthesis and depth estimation have been proposed, which do not require ground truth depth at training time. We thus make use of the term self-supervision to describe such methods: our training dataset does not explicitly expose the signal we are trying to learn. In this case the signal which is learnt is per-pixel depth.

Flynn et al. [63] introduced a novel image synthesis network called DeepStereo that generates new views by selecting pixels from nearby images. During training, the relative pose of multiple cameras is used to predict the appearance of a held-out nearby image. Then the most appropriate depths are selected to sample colours from the neighbouring images, based on plane sweep volumes. At test time, image synthesis is performed on small overlapping patches. As it requires several nearby posed images at test time, DeepStereo is not suitable for monocular depth estimation.

The Deep3D network of Xie et al. [64] also addresses the problem of novel view synthesis, where their goal is to generate the corresponding right view from an input left image (*i.e.* the source image) in the context of binocular pairs. Again using an image reconstruction loss, their method produces a distribution over all the possible disparities for each pixel. The resulting synthesized right image pixel values are a combination of the pixels on the same scan line from the left image, weighted by the probability of each disparity. The disadvantage of their image formation model, is that increasing the number of candidate disparity values in turns greatly increases the memory consumption, making it difficult to scale their approach to bigger output resolutions.

Similarly to Deep3D, Garg et al. [65] trains a network for monocular depth estimation using an image reconstruction loss. However, their image formation model is not fully differentiable. To compensate, they perform a Taylor approximation to linearise their loss, resulting in an objective that is more challenging to optimise.

Chapter 3

Multi-view Reconstruction of Highly Specular Surfaces in Uncontrolled Environments

In this chapter we present a novel method to reconstruct the surface of highly specular objects, where traditional methods relying on direct image correspondences do not work. Instead, we rely on indirect observations of the reflected environment, which allows us to compute the surface that correctly predicts the appearance of the input images.

3D reconstruction of both scenes and objects has become accessible through photogrammetry, depth cameras and affordable laser scanners. These technologies are designed with the assumption that the objects are made of diffuse materials, whose appearance does not change with the viewpoint. Consequently, objects made of highly specular, mirror-like materials cannot be reconstructed using those methods unless, they have been covered with a diffuse coating. To date, such objects can only be reconstructed in controlled environments, using expensive, tailor-made hardware such as motor controlled illumination [9, 22] or monitors displaying patterns [23].

We propose a method that accurately reconstructs mirror reflective objects in a casual setting using a consumer camera. Provided that the object is sufficiently far away from its surroundings, our method is able to reconstruct the surface of the

Figure 3.1: The method. From left to right: The initialisation from the image silhouettes, our reconstruction, a synthetic rendering of our reconstruction and a real photograph of the object in the same environment.

object based only on a panoramic image of the environment and around 30 calibrated photographs, viewing the object from different directions, as well as their silhouettes. This enables cost-effective in-the-field reconstruction of highly specular objects, such as historical artifacts that cannot be moved, let alone sprayed with diffuse paint. The comparatively small number of photographs required by our method could also facilitate fast automated quality assessment of metallic mechanical parts.

Similarly to [39], we start from the visual hull of the object's shape, and build a probability distribution of the surface normals based on correspondences between the colours reflected by the object and those in the environment. We use these distributions to form estimates of the surface normals, which are used to iteratively reconstruct the object and its details. However, we focus on highly specular objects, while their method targets objects with materials containing either a strong diffuse component and/or rough specularities. As such, we cannot rely on the roughness of the material to guide the optimisation. Although the appearance of a surface with a rough material varies smoothly as its shape changes, this is not the case for highly specular materials.

In this chapter, we present four main contributions:

- A Bayesian framework for estimating surface normals that is robust to outliers by modelling the likelihood of an observed colour as a multivariate t-distribution.

- A method to incorporate inter-reflections into the probabilistic estimate of the surface normals by explicitly modelling them.

- A method to estimate frontier points based only on the visual hull and its associated cameras.

- A publicly available dataset with both real and synthetic specular objects, captured in uncontrolled environments, and with associated ground truth geometry.

We also perform an analysis of the challenges associated with reconstructing highly specular objects in this setting. Namely, we investigate the impact of the proximity of the environments as well as their variety, the handling of inter-reflections and the sampling rate of the surface normals.

3.1 Method

We aim to reconstruct a reflective object that can be characterized by its closed surface S. We assume that the object is small enough with respect to its surroundings to treat the environment as infinitely far away. We represent the incident light field as an environment map $E(\mathbf{r})$, *i.e.* a function that maps directions to colors, (see Figure 3.2). In this setting, the color of a surface point $\mathbf{v} \in S$, observed from the direction of the eye vector $\mathbf{e}$, only depends on the surface normal $\mathbf{n}$ and the position of the camera I relative to $\mathbf{v}$. The user obtains a set of calibrated photographs of the surface, seen from different viewpoints. We use the observed color in an image to infer the possible normals at $\mathbf{v}$. If every point in the environment had a unique color, a single lookup would reveal the normal explaining $\mathbf{v}$'s color. However, similar colors might be explained by different normals. For example, a large range of normals can explain a blue sky reflection, or observing green could mean reflecting grass or

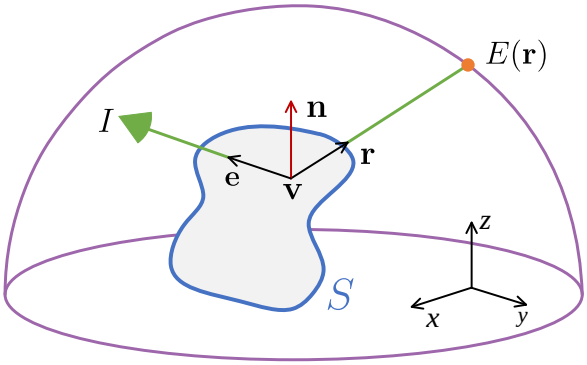


Figure 3.2: Our model. Light from the environment map $E(\mathbf{r})$ hits vertex $\mathbf{v}$ and is reflected about the unknown normal $\mathbf{n}$. The reflection follows eye vector $\mathbf{e}$ toward camera I

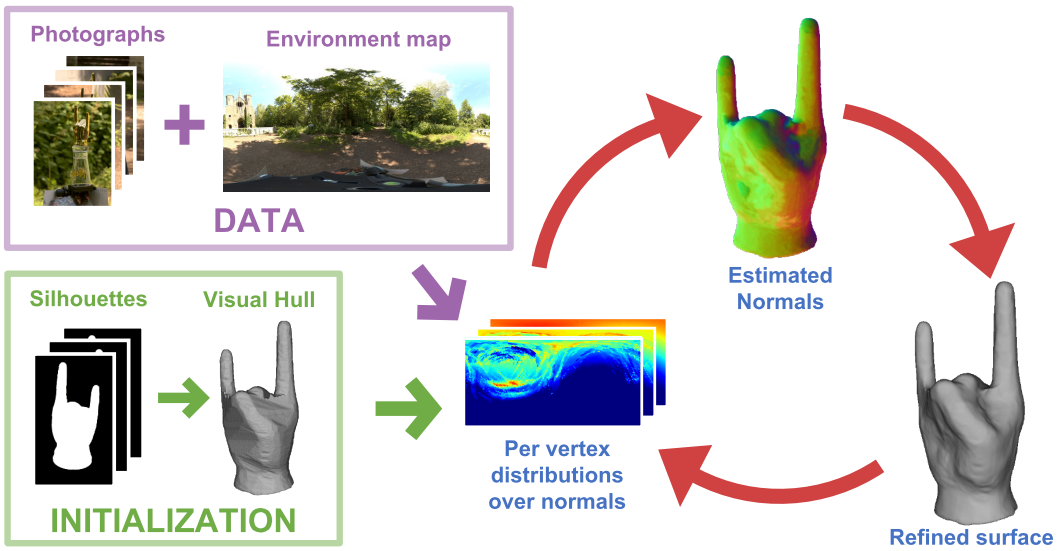


Figure 3.3: Our pipeline. The user captures multi-view images of a reflective object. The visual hull is constructed from masks of the object in each image. From this initialization and a captured environment map, our method alternates between refining the surface normals and its shape, to explain the observed reflections.

trees, introducing ambiguity. We reduce this uncertainty by combining the multiple observations of the same surface point from different viewing directions.

Overview. Our method proceeds as follows (see Figure 5.1): Our initial surface estimate S_e is a visual hull [66] formed through voxel carving from the calibrated cameras and hand made object silhouettes. A triangulated mesh of the visual hull is generated using screened Poisson surface reconstruction [67], then remeshed using [68] to ensure uniform triangle shape and size. This eliminates the need to compensate for different shapes and areas in the surface optimization. As described

in Section 3.1.1, we build a probability distribution over the possible normals for each vertex, by comparing the observed colors against the ones in the environment. We account for the estimated material's tint and inter-reflections before merging the probability distributions obtained from each of the photographs using Bayesian inference. In Section 3.1.2, we extract a representative normal for each vertex by assuming local smoothness, and then refine the mesh to better explain these normals. This procedure is then iterated until convergence. We also propose, in Section 3.1.2.2, an efficient and robust method to estimate frontier points [69, 70] on the initial mesh.

3.1.1 Estimating normal distributions

Model. Given a vertex $\mathbf{v}$ on a surface S_e, we denote $I = \{I_k\}_{k=1...K}$ as the set of cameras which $\mathbf{v}$ is visible to and $\mathbf{c}_{ok}$ as its observed color in the image from I_k, see Figure 3.2. We uniformly and densely sample the normal space S^2 to form a discrete probability distribution over each vertex's normal.

We model the conditional probability of observing $\mathbf{v}$'s color given a normal $\mathbf{n}$ in image I_k as a multivariate t-distribution t (Appendix A), centered on the color of the environment expected to be reflected from this viewpoint: $\mathbf{c}_{ek}(\mathbf{n})$. We choose the multivariate t-distribution because it is robust to outliers thanks to its heavy tails.

Given the current surface estimate, we identify two cases where reflection is impossible. First, if the normal $\mathbf{n}$ is facing away from the observer, *i.e.* if the angle between $\mathbf{n}$ and the eye vector $\mathbf{e}_k$ is larger than 90°. Second, if the reflected vector $\mathbf{r}_\mathbf{n}(\mathbf{e}_k)$ crosses the local tangent plane, *i.e.* if the angle between $\mathbf{r}_\mathbf{n}(\mathbf{e}_k)$ and the current surface normal $\mathbf{n}_m$ is larger than 90°. Interestingly, the naive solution of assigning probability zero to such normals prevents deep concavities from being recovered. To resolve this, we model the probability of observing any color as a uniform distribution $u(\mathbf{c}_{ok})$ if the normal produces an impossible reflection. Formally,

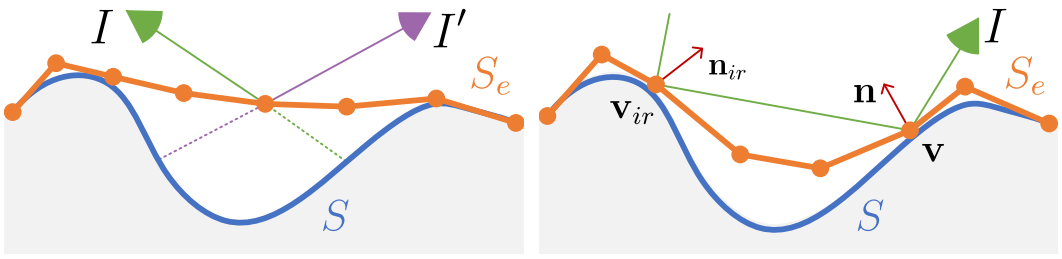
$$P(\mathbf{c}_{ok}|\mathbf{n}) = \begin{cases} t(\mathbf{c}_{ok}|\mathbf{c}_{ek}(\mathbf{n}), \nu, \sigma^2), & \text{if } \mathbf{e}_k \cdot \mathbf{n} > 0 \\ & \text{and } \mathbf{r}_{\mathbf{n}}(\mathbf{e}_k) \cdot \mathbf{n}_m > 0 \\ u(\mathbf{c}_{ok}), & \text{otherwise,} \end{cases} \tag{3.1}$$

where we set $\sigma = 0.01$, $\nu = 0.01$ in all our experiments. We define the support of the uniform distribution to be the volume $[0, \hat{c}]^3$, where $\hat{c}$ is the largest value for all of the combined color channels of the environment map, *i.e.* $u(\mathbf{c}_{ok}) = \hat{c}^{-3}$.

Modeling the probability of an observed color this way assigns high likelihoods to good samples, while not completely discarding normals producing outliers. We found that outliers fit into two main categories:

- The estimated surface originating from space carving will not exhibit interior concavities. This means that a vertex $\mathbf{v}$ far away from the true surface S is likely to give observations produced by different normals for each image, see Figure 3.4a. Similarly to [39], we alleviate this issue by not considering observations for which the angle between the current surface normal and the view direction is larger than 60°.

- When an observation is the result of an inter-reflection, see Figure 3.4b, the normal best explaining such a color will not correspond to the true surface normal. Later in this section we describe a method to explicitly model such observations.



(a) Concavities. **(b)** Inter-reflections.

Figure 3.4: Outliers. In these cases the reflected color cannot be explained by the estimated surface or the local normal alone.

We assume each observation is independent and use Bayes' rule to compute the posterior probability distributions over the normals at each vertex. That is,

$$P(\mathbf{n}_j|I) = P(\mathbf{n}_j)\prod_{k=1}^{K}\frac{P(\mathbf{c}_{ok}|\mathbf{n}_j)}{P(\mathbf{c}_{ok})}, \tag{3.2}$$

$$P(\mathbf{c}_{ok}) = \sum_j P(\mathbf{c}_{ok}|\mathbf{n}_j)P(\mathbf{n}_j). \tag{3.3}$$

We set the prior $P(\mathbf{n}_j)$ to follow a discretized von Mises-Fisher distribution centered around the current mesh normal for each vertex $\mathbf{v}$ and with concentration parameter $\kappa = 25$ for both real-world and synthetic data (Appendix A). As can be seen in Figure 3.6, the influence of the prior decreases with the number of observations.

Material estimation. We model the surface material as a tinted perfect mirror with no diffuse component. In other words, we express the color $\mathbf{c}_{ek}(\mathbf{n})$ reaching the observer as the reflected color from the environment map E scaled by an RGB triplet ρ. Formally,

$$\mathbf{c}_{ek}(\mathbf{n}) = \rho E(\mathbf{r}_{\mathbf{n}}(\mathbf{e}_k)), \tag{3.4}$$

where $\mathbf{r}_{\mathbf{n}}(\mathbf{e}_k)$ is the reflection of the eye vector $\mathbf{e}_k$ about a normal $\mathbf{n}$ at vertex $\mathbf{v}$. The Fresnel effect is significant only for observations at glancing angles, as we discard such observations we do not incorporate it in our material model.

To compute the material tint, we use the estimated surface to obtain the colors $E(\mathbf{r}_{\mathbf{n}}(\mathbf{e}_k))$ for each vertex in all of the photographs. We then obtain ρ by mapping these colors to the observed ones using a robust linear regression with 20 samples and 10^4 RANSAC iterations.

Inter-reflections. We classify the observations as either *direct reflections* or *inter-reflections*. A direct reflection is the result of a single mirror reflection between the environment and the observer. In contrast, an inter-reflection is the result of two or more perfect mirror reflections on the surface of the object, see Figure 3.4b. These observations are difficult to model because instead of depending on a single surface normal, an arbitrary number of surface normals may be needed to explain them.

However, we observe that, as seen in Figure 3.5, most of the observations can

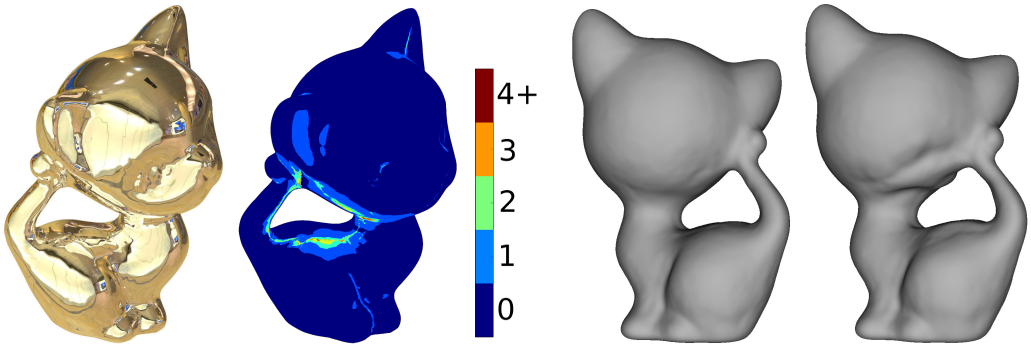


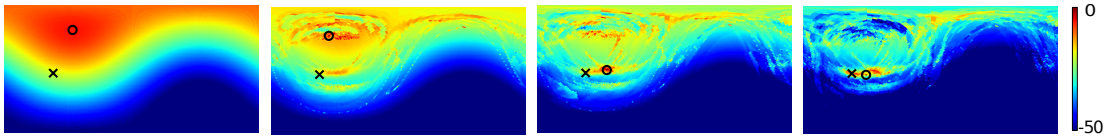
Figure 3.5: Inter-reflections. Left: We found that the vast majority of observations are the results of either direct reflections (dark blue) or single-bounce inter-reflections (light blue). **Right:** Ignoring inter-reflections causes artifacts (right) not present in our reconstructions (left)

be modeled as either a direct reflection (*dr*) or a single bounce inter-reflection (*ir*). Based on this observation, we focus on the first level of inter-reflection only. A simple but naive approach is to ignore an observation if it would be an inter-reflection given the current estimate of the surface. This would however be a waste of information. We instead choose to augment $\mathbf{c}_{ek}$ to explicitly take inter-reflections into account by ray-tracing the estimated surface S_e and computing the colors resulting from one bounce reflections. In other words,

$$\mathbf{c}_{ek}(\mathbf{n}) = \begin{cases} \rho E(\mathbf{r}_{\mathbf{n}}(\mathbf{e}_k)) & (dr) \\ \rho^2 E(\mathbf{r}_{\mathbf{n}_{ir}(k)}(\mathbf{r}_{\mathbf{n}}(\mathbf{e}_k))) & (ir), \end{cases} \tag{3.5}$$

where $\mathbf{n}_{ir}(k)$ is the normal where the reflected ray intersects with the mesh again (see Figure 3.4b).

Even though this method is an approximation because of the use of the estimated surface S_e, its accuracy increases with the number of iterations as the surface gets refined. Figure 3.5 shows that it significantly improves the results and that an artifact-free reconstruction would not be possible without explicitly handling inter-reflections.



(a) Prior **(b)** 3 observations **(c)** 6 observations **(d)** 9 observations

Figure 3.6: The evolution of the probability distribution (seen in log-space) over the possible normals for a vertex. The prior is a von Mises-Fisher distribution centered on the current surface normal. As the vertex is observed in more images, we see that the distribution becomes more confident and its mode (the circle) approaches the ground truth normal (the cross).

3.1.2 Optimization

We use the estimated probability distributions for the surface normals to drive the shape of the object. Taking heed of the experimental evaluation performed by Hernandez *et al.* [71], we do not directly refine the mesh based on the probability distributions, but adopt a two-step process instead.

We first extract representative normals $\mathbf{n}_i$ for each vertex $\mathbf{v}_i$ on the surface while enforcing local smoothness (Section 3.1.2.1). We then refine the surface to better explain these normals (Section 3.1.2.2). To adequately constrain this problem, we make use of frontier points (Section 3.1.2.3) whose positions are guaranteed to match those of the original object. We iterate this process until convergence, making use the refined surface at each iteration to produce more confident probability distributions for the surface normals.

3.1.2.1 Representative normals

We fix the location of all vertices and model the extraction of a representative normal $\mathbf{n}_i$ for every vertex $\mathbf{v}_i$ as an energy minimization problem. Its objective function consists of two distinct parts,

$$E_{\mathrm{n}} = \alpha_{\mathrm{d}} E_{\mathrm{d}} + \alpha_{\mathrm{s}} E_{\mathrm{s}}, \tag{3.6}$$

where E_{d} is the data term and E_{s} is the smoothness term. We set $\alpha_{\mathrm{d}} = 1$, $\alpha_{\mathrm{s}} = 5 \times 10^5$ and initialize each normal to the global maximum of its corresponding distribution in all our experiments.

Data term. The first term penalizes normals that have low probability according to the estimated distributions. Specifically,

$$E_d = \sum_i |\log P(\mathbf{n}_i)|^2 . \tag{3.7}$$

As the distributions of the normals are discretized, we remodel the continuous density in log-space by placing von Mises-Fisher distributions at each discrete probability sample $P(n_p)$. Namely,

$$\log P(\mathbf{n}_i) \propto \sum_p \log P(\mathbf{n}_p) e^{\kappa \mathbf{n}_p^T \mathbf{n}_i}, \tag{3.8}$$

where we set $\kappa = \theta_d^{-2}$. For large values of κ, the angle θ_d roughly corresponds to one standard deviation (Appendix A). We initialize θ_d to 5° and linearly decrease it to reach 2°, its minimum value, at the fifth iteration of the entire optimization process.

Smoothness term. The second term ensures local smoothness of the normals, while not penalizing curvature. This formulation avoids the caveats of a simple per-edge dot product, which would only yield zero cost for flat surfaces. Instead our cost depends on the mean of the normal vectors in the one-ring neighborhood. In other words,

$$E_\mathrm{s} = \sum_i |1 - \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \mathbf{n}_i \cdot \mathbf{n}_j|^2, \tag{3.9}$$

where $\mathcal{N}_i$ is the one-ring neighborhood of $\mathbf{v}_i$.

3.1.2.2 Surface refinement

We refine the surface to better explain the now fixed representative normals. Again, we model this problem as the minimization of an objective function consisting of three distinct parts. Namely,

$$E_m = \alpha_\mathrm{mesh} E_\mathrm{mesh} + \alpha_\mathrm{v} E_\mathrm{v} + \alpha_\mathrm{fp} E_\mathrm{fp}, \tag{3.10}$$

where E_{mesh} is the normal term, E_{v} is the volume term and E_{fp} is the frontier point term. We set $\alpha_{\text{mesh}} = 10$, $\alpha_{\text{v}} = 1$ and $\alpha_{\text{fp}} = 1$ in all our experiments. For these parameters to work across multiple scales we resize the mesh for the surface refinement so that the average edge size is unit length. Furthermore, inspired by [72] we only allow for vertices to be displaced along their representative normals during the optimization to minimize the occurrence of self-intersecting triangles.

Normal term. For the purpose of matching the mesh to the optimized normals we employ the linear cost E_{mesh} from [72]. This cost enforces the edges in the one-ring neighborhood $\mathcal{N}_i$ around each vertex $\mathbf{v}_i$ to be perpendicular to its estimated normal $\mathbf{n}_i$. Formally,

$$E_{mesh} = \sum_i \sum_{j,k \in \mathcal{N}_i} |\mathbf{n}_i \cdot (\mathbf{v}_j - \mathbf{v}_k)|^2. \tag{3.11}$$

Volume term. We introduce a penalty to prevent the mesh from increasing in volume compared to the visual hull. We define this penalty as

$$E_v = \sum_i |\max(0, \mathbf{n}_i \cdot (\mathbf{v}_i - \bar{\mathbf{v}}_i))|^2, \tag{3.12}$$

where $\bar{\mathbf{v}}_i$ is the vertex closest to $\mathbf{v}_i$ on the visual hull.

Frontier point term. In epipolar geometry *contour generators* [69, 70] are the set of 3D curves, one per camera, where the ground truth surface coincides with the visual hull. Each of these curves projects onto the silhouette contour in its associated image. Points which lie at the intersection of two contour generators are *frontier points*. If found, such points can be used as unbiased and strong constraints in the surface optimization to anchor the location of vertices and avoid drift. Based on this, we penalize vertices close to frontier points for deviating from their initial visual hull positions. In other words,

$$E_{fp} = \sum_i ||\mathbf{v}_i - \bar{\mathbf{v}}_i||^2 e^{-r_i^2/2\sigma_{fp}^2}, \tag{3.13}$$

where r_i is the distance from $\mathbf{v}_i$ to the closest point in the set of frontier points and $\sigma_{fp} = 3$.

3.1.2.3 Estimating frontier points

As our initial surface estimate is in the form of a visual hull, we cannot rely on its contours alone to identify frontier points. Indeed, points on the visual hull that project onto the contour of a given image I form a superset of the associated contour generator. For smooth surfaces, a frontier point $\mathbf{v}$ associated with two cameras I and I' has its normal orthogonal to the epipolar plane defined by the triplet $(\mathbf{v}, I, I')$. A naive method would identify a vertex as a frontier point if it lies on the contours in two images and if its normal is orthogonal to the corresponding epipolar plane. However, we found this method unreliable and prone to false positives due to approximate contours and inaccurate initial normals.

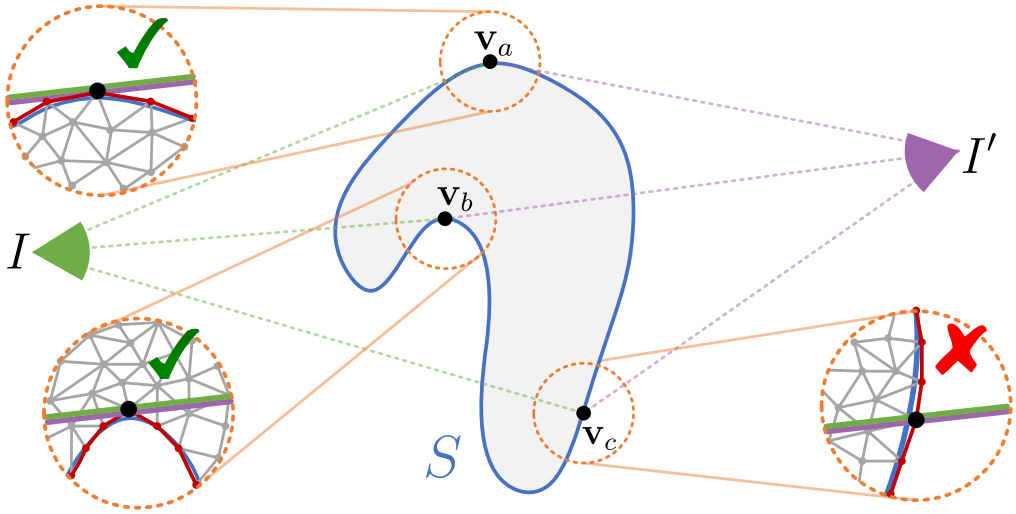


Figure 3.7: Frontier points. The vertices $\mathbf{v}_a, \mathbf{v}_b$ and $\mathbf{v}_c$ all lie on the contours in the images taken from I and I'. Only $\mathbf{v}_c$ is not identified as a frontier point as the epipolar plane (the purple-green line) crosses the union of the projective neighborhoods $\mathcal{N}_c(I)$ and $\mathcal{N}_c(I')$ (highlighted in red).

Instead, we observe that a plane intersecting the surface at a vertex $\mathbf{v}$, but no other point in its neighborhood, is a tangent plane at $\mathbf{v}$. Based on this observation, we identify a vertex $\mathbf{v}_i$ that lies on the contour in two images I and I' as a frontier point if its neighborhood does not cross the epipolar plane (Figure 3.7). To cope with inaccurate contours, we let the neighborhood of $\mathbf{v}_i$ depend on I and I'. Specifically, we define the neighborhood of $\mathbf{v}_i$ as the union of the *projective neighborhoods* $\mathcal{N}_i(I)$ and $\mathcal{N}_i(I')$, where $\mathcal{N}_i(I)$ is the set of points that lie on the contour from I and

are inside a narrow cone with radius r_{fp} at $\mathbf{v}_i$. We set r_{fp} to be 7.5 times the average edge length in all our experiments.

Post-processing Once the surface has been refined, the assumption of a uniform triangulation no longer holds. Indeed, our energy terms do not prevent irregular triangle shapes and inverted faces. Modifying the terms to prevent such issues introduces more parameters and did not converge to the right solution in our experiments. Instead, we correct the issues in a post-processing step identical to how we obtain the triangulated mesh from the visual hull.

3.2 Implementation and results

Data capture We generated our synthetic dataset by rendering 36 views of four different objects, KITTY, BUNNY, PLANCK and ROCKERARM. Each object was rendered using three different high-dynamic-range (HDR) environment maps TOKYO, PAPERMILL and LOFT, all freely available online [73], as can be seen in Figure 3.11. To investigate the effect of breaking the assumption that the environment is infinitely far away, we also rendered the objects in the 3D scene SPONZA using a global illumination path tracer.

Our real-world dataset contains three objects, PIGGY, TEDDY and HEAVYMETAL captured in two scenes, THEATER and CHURCH. Capturing all objects in one scene took approximately 50 minutes on site. We obtained the images for this dataset by merging bracketed DSLR photographs into linear HDR images. The environment map was obtained by stitching together wide-angle photographs. We manually created the silhouette images using standard image editing software, which took roughly one hour per object. We also acquired accurate geometry for PIGGY and TEDDY by coating them with diffusing spray and laser scanning them. The camera extrinsics and intrinsics were recovered using a standard structure from motion package [74] and we attached texture rich postcards to the tripod supporting the objects to ensure good calibration. We aligned the environment map and the structure from motion coordinate space in less than ten minutes by manually locating four point correspondences in the environment map and the background of the object's

photographs and then applying Kabsch's algorithm [75] to align them.

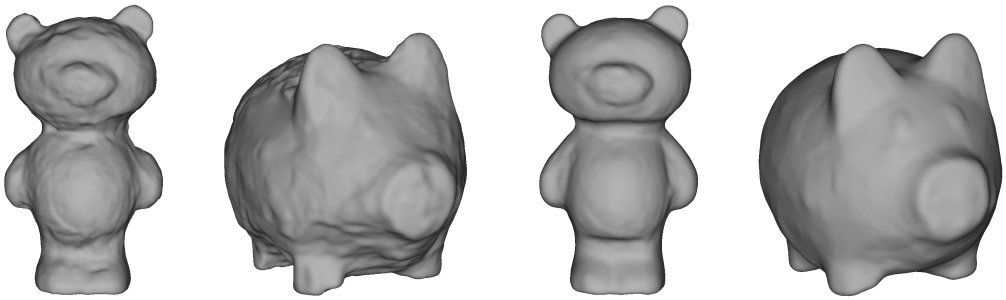
Optimization We use the HEALpix projection [76] to uniformly sample the normal space S^2 with $N_p = 12 \times 4^{N_r}$ samples. Unless mentioned otherwise we use the resolution exponent $N_r = 5$ and $N_p = 12288$ samples. We downsample the environment map to match the sampling density of S^2. The distributions are computed on a GPU and the Optix Prime ray tracing library [77] is used to resolve visibilities and inter-reflections.

We use the Ceres Solver library [78] implementation of the L-BFGS algorithm to both extract representative normals and refine the surface estimate. The optimizations are run with an unbounded number of iterations until the convergence criterion $d\mathbf{E}/\mathbf{E} < 10^{-6}$ is met. In the pre- and post-processing stages we remesh all our surfaces to contain 50000 vertices.

3.2.1 Synthetic results

To test our method and validate the assumptions it relies on, we ran a set of experiments using our synthetic dataset. See Figure 3.12 for qualitative results of our method on a cross section of the synthetic dataset.

Environments and inter-reflections In Figure 3.10 we see that the choice of environment only weakly affects the reconstruction quality. In the same figure, we also analyze the impact of inter-reflections on the reconstruction quality. As can be seen from the convergence plots, ignoring inter-reflections often causes the reconstruc-



(a) Our interpretation of [39] **(b)** Our method

Figure 3.8: Reconstructions of TEDDY and PIGGY in THEATER after 5 iterations of either our original method or a modified version that uses [39] to compute the distributions over the surface normals.

tion to diverge. Discarding observations that would be inter-reflections based the current surface estimate wastes useful data and does not reach the reconstruction quality of our explicit method.

Proximity To evaluate the impact of the object's proximity to the environment on the reconstruction quality, we rendered two new versions of BUNNY in SPONZA. One where it is twice as large and one where it is twice as small. As the size of the scene remains constant, the visual angle of the object in the images increases with the size of the object as the observer is unable to move further away. Intuitively, the reconstruction of the larger object should be of lower quality as the reflections disagree more with the environment map due to parallax. In our other experiments, the visual angle of BUNNY is 10°. As expected, the reconstruction error of the small version (5°) is 7.4% lower while the error increases by 8.3% for the large version (20°).

Resolution Table 3.1 shows that the final reconstruction error is not heavily affected by the choice of the sampling density of the normal space S^2. In our other experiments, we use $N_r = 5$ as it is a good compromise between speed and quality.

Resolution exponent N_r					
2	3	4	5	6	7
-21.74%	-31.22%	-31.24%	-31.86%	-32.58%	**-32.8%**

Table 3.1: The decrease in RMS error compared to the visual hull for BUNNY in SPONZA when varying N_r for the sampling density of S^2.

Number of input images To evaluate the impact of the number of input images on the final reconstruction quality, we ran an experiment where we decrease the amount of input images. As we can see in Figure 3.9 the reconstruction error improves significantly as more images are used but we also see diminishing returns.

3.2.2 Real-world results

We also evaluate our method on a real-world dataset, see Figure 3.13 for qualitative results. Despite the challenges of real-world capture, our method accurately reconstructs the objects. Our method recovers deep concavities such as the palm of HEAVYMETAL and fine details such as the eyes of PIGGY. Results from the

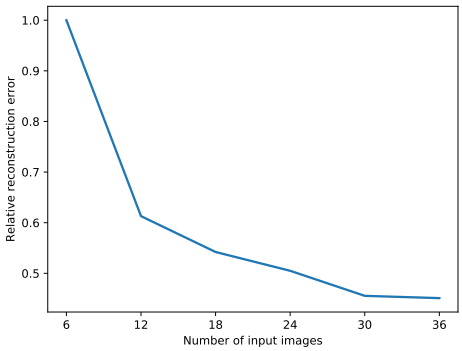


Figure 3.9: Impact of the number of images. The relative RMS error of the reconstruction of BUNNY in SPONZA when varying the number of input images from 6 to 36.

CHURCH scene show that it is hard to recover from false positive frontier points. Scene specific parameters could alleviate this, but we used the same parameters in all experiments. See Figure 3.1 for our reconstruction of TEDDY in the INCEPTION scene.

Comparison with [39]. Although Oxholm and Nishino [39] do not explicitly target highly specular objects, their BRDF model does in theory support this case. Unfortunately, we cannot directly compare our methods as they were unable to run their method on new datasets and it is not possible to run our method on their dataset due to poorly calibrated images.

Instead, we modified our method to compute the distributions over the surface normals using their algorithm, keeping the rest of our pipeline identical. This compromise was made as using their area priors during surface refinement led to worse reconstructions (Appendix B). Figure 3.8 shows that this introduces bumps on the reconstructed surfaces, because their method is sensitive to outliers without a BRDF that smooths out the high frequency components in the environment.

3.3 Conclusion

We have presented a method to accurately reconstruct the shape of highly specular objects in uncontrolled environments from a small number of photographs. To our knowledge, our method is the first to achieve this using only commodity hardware. We presented a bayesian framework to estimate surface normals by predicting the

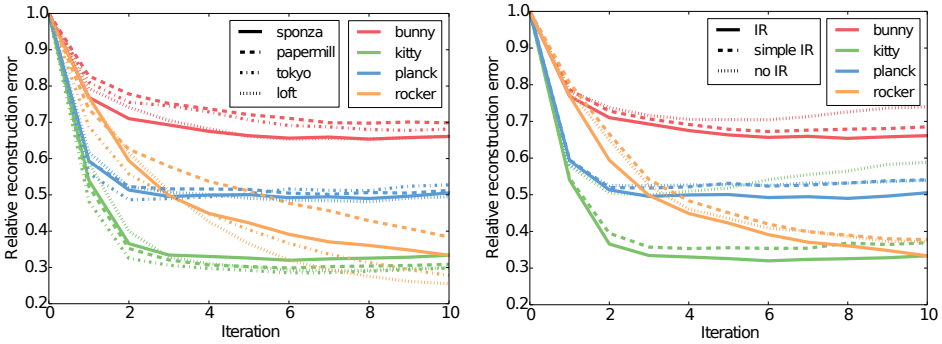


Figure 3.10: Left. The RMS error relative to that of the initial visual hull for four synthetic objects, each rendered in four environments. Our method is robust to varied environments and converges in under 10 iterations. **Right.** The RMS error relative to that of the visual hull for different strategies that deal with inter-reflections: Ignoring inter-reflections all together (no IR), discarding observations that inter-reflect given the estimated surface (simple IR) and our explicit method (IR).

Figure 3.11: The panoramic image of the four synthetic environments we used in our experiments, as well as the two environments in which we captured our real-world objects.

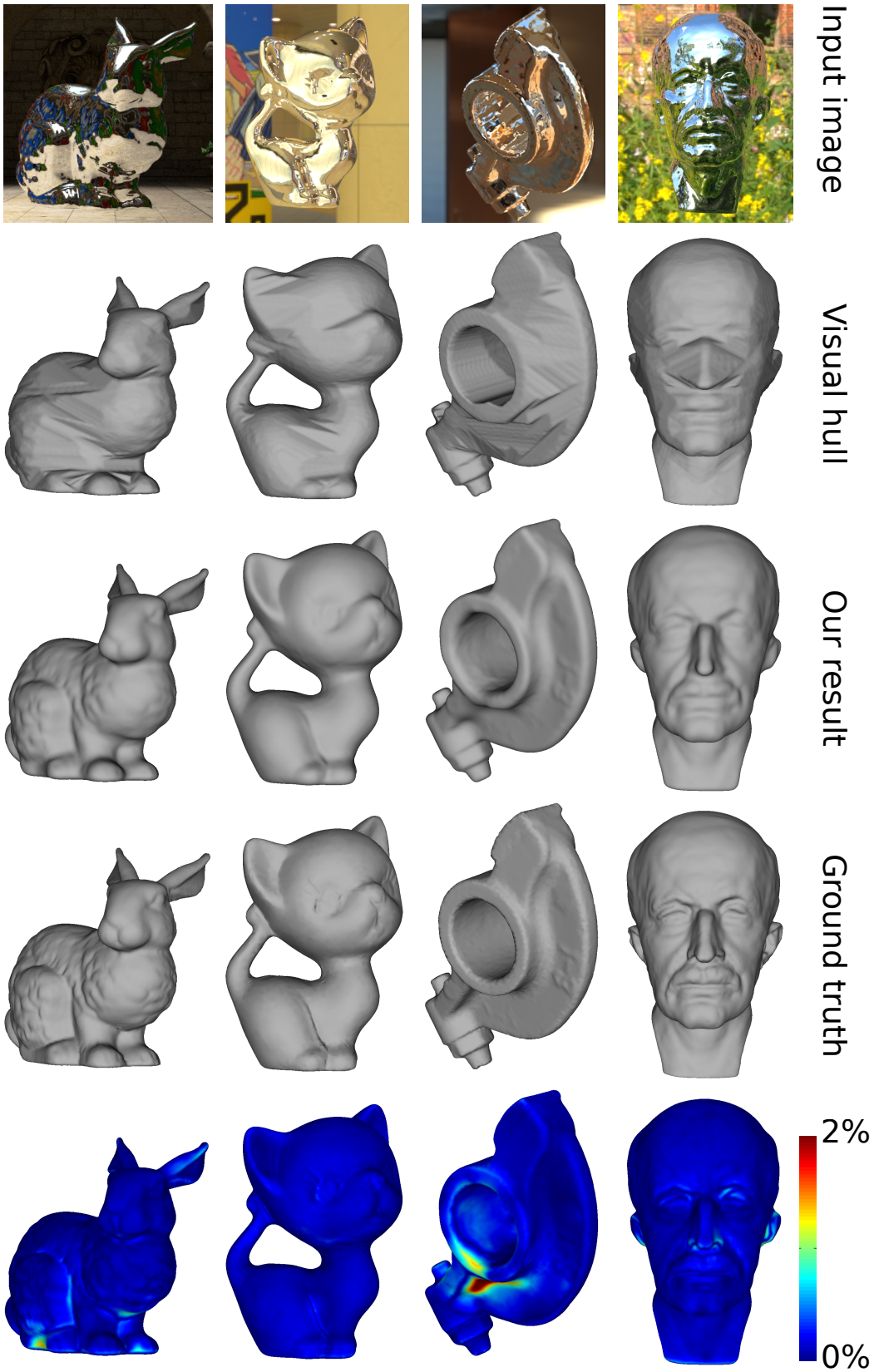


Figure 3.12: Our reconstructions of synthetic objects compared to one of the input photographs and the ground truth mesh. **Left to right:** BUNNY in the SPONZA scene, KITTY in TOKYO, ROCKERARM in LOFT and PLANCK in PAPERMILL. All objects were reconstructed from 36 input photographs. The last row shows the reconstruction errors relative to the bounding box diagonals.

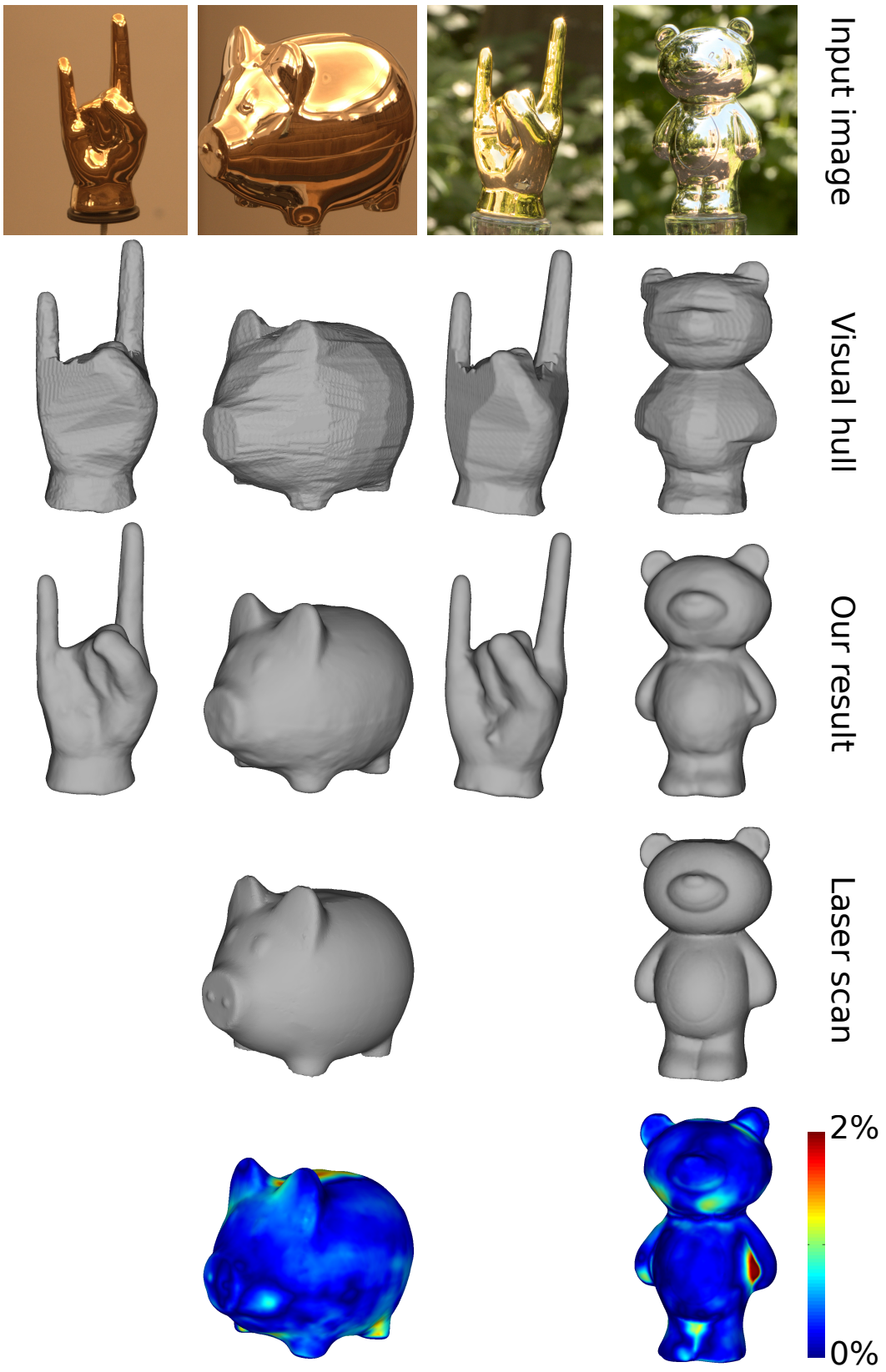


Figure 3.13: Our reconstructions of real-world objects compared to an input photograph and a laser scan (where applicable). **Left to right:** HEAVYMETAL and PIGGY in the THEATER scene (35 and 25 input photographs respectively), HEAVYMETAL and TEDDY in the CHURCH scene (27 and 26 input photographs respectively). The last row shows the reconstruction errors relative to the bounding box diagonals.

appearance of the surface for all possible normals given a panoramic image of the environment and by modelling inter-reflections. We also released a publicly available dataset of both real and synthetic specular objects, captured in uncontrolled environments alongside their ground truth geometry, to encourage comparison and future work. In the next Chapter we will focus on another end of the image-based reconstruction spectrum and look at depth prediction from single images.

Chapter 4

Self-Supervised Monocular Depth Estimation with Left-Right Consistency

In this chapter we focus on 3D reconstruction from a single input image. We make use of a large number of stereo pairs, which are easy to capture. In contrast from Chapter 3, where we used multiple images to do indirect matching and reconstruct specular surfaces, we train a neural network to regress the depth of an input image. We do so by matching indirectly, at training time, by predicting the appearance of the second image in a stereo pair.

Depth estimation from images has a long history in computer vision. Fruitful approaches have relied on structure from motion, shape-from-X, binocular, and multi-view stereo. However, most of these techniques rely on the assumption that multiple observations of the scene of interest are available. These can come in the form of multiple viewpoints, or observations of the scene under different lighting conditions. To overcome this limitation, there has recently been a surge in the number of works that pose the task of monocular depth estimation as a supervised learning problem [55, 57, 54]. These methods attempt to directly predict the depth of each pixel in an image, using models that have been trained offline on large collections of ground truth depth data. While these methods have enjoyed great success, to date they have been restricted to scenes where large image collections and their

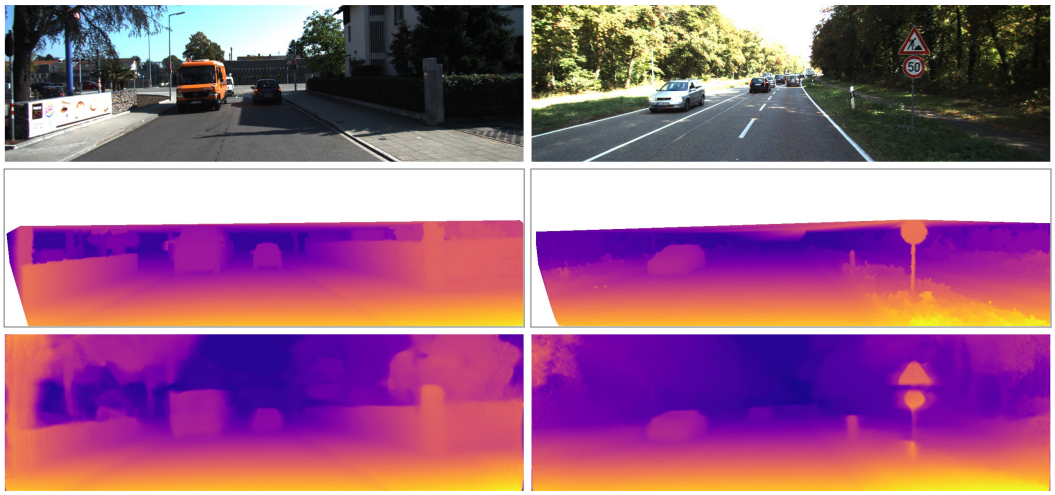


Figure 4.1: Our depth prediction results on KITTI 2015. Top to bottom: input image, ground truth disparities, and our result. Our method is able to estimate depth for thin structures such as street signs and poles.

corresponding pixel depths are available.

Understanding the shape of a scene from a single image, independent of its appearance, is a fundamental problem in machine perception. There are many applications such as synthetic object insertion in computer graphics [79], synthetic depth of field in computational photography [80], grasping in robotics [81], using depth as a cue in human body pose estimation [82], robot assisted surgery [83], and automatic 2D to 3D conversion in film [64]. Accurate depth data from one or more cameras is also crucial for self-driving cars, where expensive laser-based systems are often used.

Humans perform well at monocular depth estimation by exploiting cues such as perspective, scaling relative to the known size of familiar objects, appearance in the form of lighting and shading and occlusion [84]. This combination of both top-down and bottom-up cues appears to link full scene understanding with our ability to accurately estimate depth. In this work, we take an alternative approach and treat automatic depth estimation as an image reconstruction problem during training. Our fully convolutional model does not require any depth data, and is instead trained to synthesise depth as an intermediate. It learns to predict the pixel-level correspondence between pairs of rectified stereo images that have a known camera baseline. There are some existing methods that also address the same problem,

but with several limitations. For example they are not fully differentiable, making training suboptimal [65], or have image formation models that do not scale to large output resolutions [64]. We improve upon these methods with a novel training objective and enhanced network architecture that significantly increases the quality of our final results. An example result from our algorithm is illustrated in Fig. 4.1. Our method is fast and only takes on the order of 35 milliseconds to predict a dense depth map for a 512×256 image on a modern GPU. Specifically, we propose the following contributions:

- A network architecture that performs end-to-end unsupervised monocular depth estimation with a novel training loss that enforces left-right depth consistency inside the network.
- An evaluation of several training losses and image formation models highlighting the effectiveness of our approach.
- In addition to showing state of the art results on a challenging driving dataset, we also show that our model generalises to three different datasets, including a new outdoor urban dataset that we have collected ourselves, which we make openly available.

4.1 Method

This section describes our single image depth prediction network. We introduce a novel depth estimation training loss, featuring an inbuilt left-right consistency check, which enables us to train on image pairs without requiring supervision in the form of ground truth depth.

4.1.1 Depth Estimation as Image Reconstruction

Given a single image I at test time, our goal is to learn a function f that can predict the per-pixel scene depth, $\hat{d} = f(I)$. Most existing learning based approaches treat this as a supervised learning problem, where they have color input images and their corresponding target depth values at training. It is presently not practical to acquire

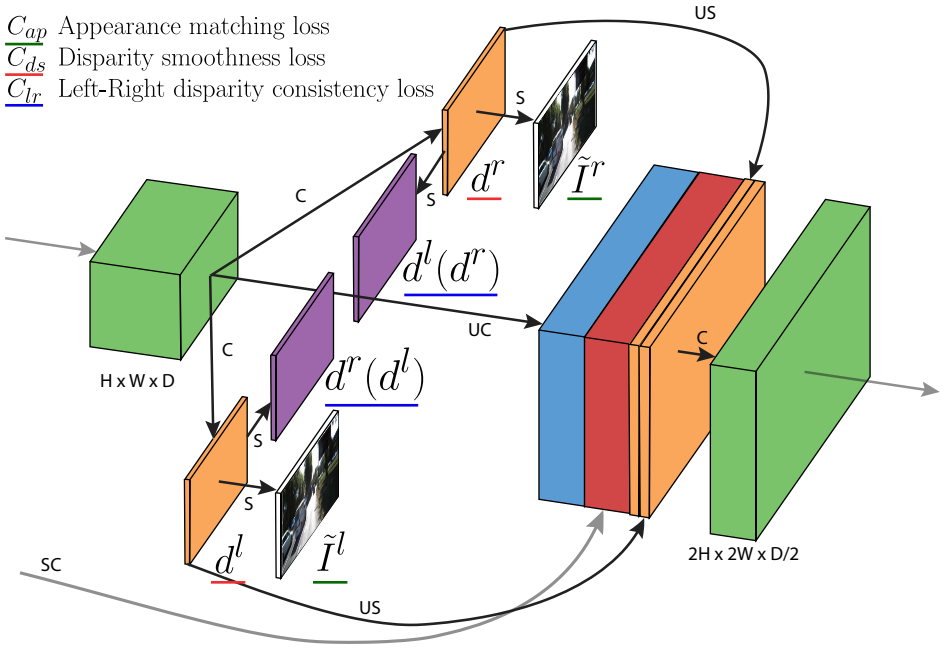


Figure 4.2: Our loss module outputs left and right disparity maps, d^l and d^r. The loss combines smoothness, reconstruction, and left-right disparity consistency terms. This same module is repeated at each of the four different output scales. C: Convolution, UC: Up-Convolution, S: Bilinear Sampling, US: Up-Sampling, SC: Skip Connection.

such ground truth depth data for a large variety of scenes. Even expensive hardware, such as laser scanners, can be imprecise in natural scenes featuring movement and reflections. As an alternative, we instead pose depth estimation as an image reconstruction problem during training. The intuition here is that, given a calibrated pair of binocular cameras, if we can learn a function that is able to reconstruct one image from the other, then we have learned something about the 3D shape of the scene that is being imaged.

Specifically, at training time, we have access to two images I^l and I^r, corresponding to the left and right color images from a calibrated stereo pair, captured at the same moment in time. Instead of trying to directly predict the depth, we attempt to find the dense correspondence field d^r that, when applied to the left image, would enable us to reconstruct the right image. We will refer to the reconstructed image $I^l(d^r)$ as $\tilde{I}^r$. Similarly, we can also estimate the left image given the right one, $\tilde{I}^l = I^r(d^l)$. Assuming that the images are rectified [85], d corresponds to the image disparity - a scalar value per pixel that our model will learn to predict. Given

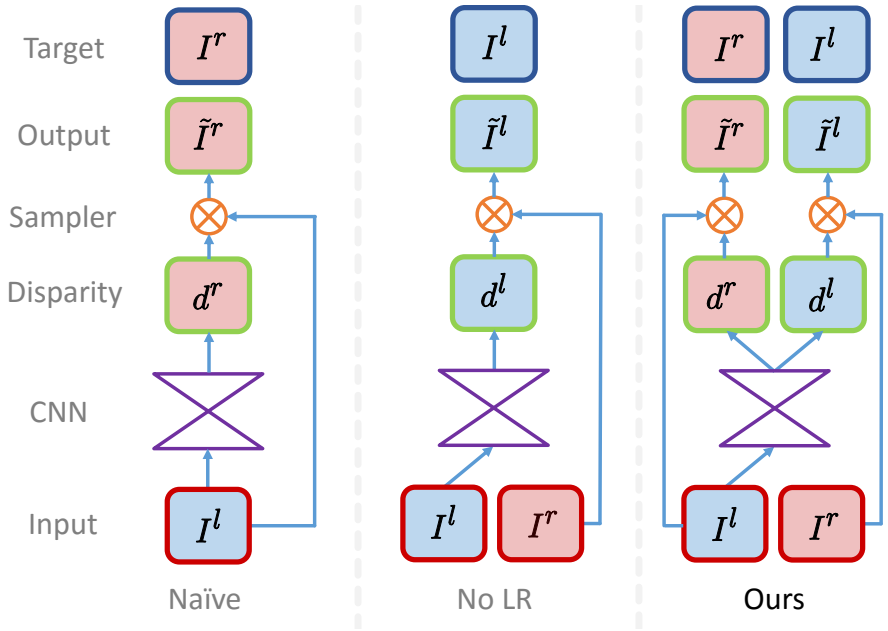


Figure 4.3: Sampling strategies for backward mapping. With naïve sampling the CNN produces a disparity map aligned with the target instead of the input. No LR corrects for this, but suffers from artifacts. Our approach uses the left image to produce disparities for both images, improving quality by enforcing mutual consistency.

the baseline distance b between the cameras and the camera focal length f, we can then trivially recover the depth $\hat{d}$ from the predicted disparity, $\hat{d} = bf/d$.

4.1.2 Depth Estimation Network

At a high level, our network estimates depth by inferring the disparities that warp the left image to match the right one. The key insight of our method is that we can simultaneously infer both disparities (left-to-right and right-to-left), using only the left input image, and obtain better depths by enforcing them to be consistent with each other.

Our network generates the predicted image with backward mapping using a bilinear sampler, resulting in a fully differentiable image formation model. As illustrated in Fig. 4.3, naïvely learning to generate the right image by sampling from the left one will produce disparities aligned with the right image (target). However, we want the output disparity map to align with the input left image, meaning the network has to sample from the right image. We could instead train the network to generate the left view by sampling from the right image, thus creating a left view

aligned disparity map (**No LR** in Fig. 4.3). While this alone works, the inferred disparities exhibit 'texture-copy' artifacts and errors at depth discontinuities as seen in Fig. 4.5. We solve this by training the network to predict the disparity maps for both views by sampling from the opposite input images. This still only requires a single left image as input to the convolutional layers and the right image is only used during training (**Ours** in Fig. 4.3). Enforcing consistency between both disparity maps using this novel left-right consistency cost leads to more accurate results.

Our fully convolutional architecture is inspired by DispNet [48], but features several important modifications that enable us to train without requiring ground truth depth. Our network, is composed of two main parts - an encoder (from cnv1 to cnv7b) and decoder (from upcnv7), please see the supplementary material for a detailed description. The decoder uses skip connections [49] from the encoder's activation blocks, enabling it to resolve higher resolution details. We output disparity predictions at four different scales (disp4 to disp1), which double in spatial resolution at each of the subsequent scales, as can be seen in Figure 4.1. Even though it only takes a single image as input, our network predicts two disparity maps at each output scale - left-to-right and right-to-left.

4.1.3 Training Loss

We define a loss C_s at each output scale s, forming the total loss as the sum $C = \sum_{s=1}^{4} C_s$. Our loss module (Fig. 4.2) computes C_s as a combination of three main terms,

$$C_s = \alpha_{ap}(C_{ap}^l + C_{ap}^r) + \alpha_{ds}(C_{ds}^l + C_{ds}^r) + \alpha_{lr}(C_{lr}^l + C_{lr}^r), \tag{4.1}$$

where C_{ap} encourages the reconstructed image to appear similar to the corresponding training input, C_{ds} enforces smooth disparities, and C_{lr} prefers the predicted left and right disparities to be consistent. Each of the main terms contains both a left and a right image variant, but only the left image is fed through the convolutional layers.

Next, we present each component of our loss in terms of the left image (*e.g.* C_{ap}^l). The right image versions, *e.g.* C_{ap}^r, require to swap left for right and

"Encoder"						
layer	**k**	**s**	**chns**	**in**	**out**	**input**
cnv1	7	2	3/32	1	2	left
cnv1b	7	1	32/32	2	2	cnv1
cnv2	5	2	32/64	2	4	cnv1b
cnv2b	5	1	64/64	4	4	cnv2
cnv3	3	2	64/128	4	8	cnv2b
cnv3b	3	1	128/128	8	8	cnv3
cnv4	3	2	128/256	8	16	cnv3b
cnv4b	3	1	256/256	16	16	cnv4
cnv5	3	2	256/512	16	32	cnv4b
cnv5b	3	1	512/512	32	32	cnv5
cnv6	3	2	512/512	32	64	cnv5b
cnv6b	3	1	512/512	64	64	cnv6
cnv7	3	2	512/512	64	128	cnv6b
cnv7b	3	1	512/512	128	128	cnv7

"Decoder"						
layer	**k**	**s**	**chns**	**in**	**out**	**input**
upcnv7	3	2	512/512	128	64	cnv7b
icnv7	3	1	1024/512	64	64	upcnv7+cnv6b
upcnv6	3	2	512/512	64	32	icnv7
icnv6	3	1	1024/512	32	32	upcnv6+cnv5b
upcnv5	3	2	512/256	32	16	icnv6
icnv5	3	1	512/256	16	16	upcnv5+cnv4b
upcnv4	3	2	256/128	16	8	icnv5
icnv4	3	1	128/128	8	8	upcnv4+cnv3b
disp4	3	1	128/2	8	8	icnv4
upcnv3	3	2	128/64	8	4	icnv4
icnv3	3	1	130/64	4	4	upcnv3+cnv2b+disp4*
disp3	3	1	64/2	4	4	icnv3
upcnv2	3	2	64/32	4	2	icnv3
icnv2	3	1	66/32	2	2	upcnv2+cnv1b+disp3*
disp2	3	1	32/2	2	2	icnv2
upcnv1	3	2	32/16	2	1	icnv2
icnv1	3	1	18/16	1	1	upcnv1+disp2*
disp1	3	1	16/2	1	1	icnv1

Table 4.1: Our network architecture, where **k** is the kernel size, **s** the stride, **chns** the number of input and output channels for each layer, **in**put and **out**put is the downscaling factor for each layer relative to the input image, and **input** corresponds to the input of each layer where $+$ is a concatenation and $*$ is a $2\times$ upsampling of the layer.

Method	Dataset	Abs Rel	Sq Rel	RMSE	RMSE log	*D1-all*	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Ours with Deep3D [64]	K	0.412	16.37	13.693	0.512	66.85	0.690	0.833	0.891
Ours with Deep3Ds [64]	K	0.151	1.312	6.344	0.239	59.64	0.781	0.931	0.976
Ours No LR	K	0.123	1.417	6.315	0.220	30.318	0.841	0.937	0.973
Ours	K	0.124	1.388	6.125	0.217	30.272	0.841	0.936	0.975
Ours	CS	0.699	10.060	14.445	0.542	94.757	0.053	0.326	0.862
Ours	CS + K	0.104	1.070	5.417	0.188	25.523	0.875	0.956	0.983
Ours pp	CS + K	0.100	0.934	5.141	0.178	25.077	0.878	0.961	**0.986**
Ours resnet pp	CS + K	**0.097**	**0.896**	**5.093**	**0.176**	**23.811**	**0.879**	**0.962**	**0.986**
Ours Stereo	K	0.068	0.835	4.392	0.146	9.194	0.942	0.978	0.989
		Lower is better				Higher is better			

Table 4.2: Comparison of different image formation models. Results on the KITTI 2015 stereo 200 training set disparity images [86]. For training, K is the KITTI dataset [86] and CS is Cityscapes [87]. Our model with left-right consistency performs the best, and is further improved with the addition of the Cityscapes data. The last row shows the result of our model trained *and tested* with two input images instead of one (see Sec. 4.2.3).

to sample in the opposite direction.

Appearance Matching Loss During training, the network learns to generate an image by sampling pixels from the opposite stereo image. Our image formation model uses the image sampler from the spatial transformer network (STN) [88] to sample the input image using a disparity map. The STN uses bilinear sampling where the output pixel is the weighted sum of four input pixels. In contrast to alternative

approaches [65, 64], the bilinear sampler used is locally fully differentiable and integrates seamlessly into our fully convolutional architecture. This means that we do not require any simplification or approximation of our cost function.

Inspired by [89], we use a combination of an $L1$ and single scale SSIM [90] term as our photometric image reconstruction cost C_{ap}, which compares the input image I^l_{ij} and its reconstruction $\tilde{I}^l_{ij}$, where N is the number of pixels,

$$C^l_{ap} = \frac{1}{N}\sum_{i,j} \alpha \frac{1 - \text{SSIM}(I^l_{ij}, \tilde{I}^l_{ij})}{2} + (1-\alpha)\left\| I^l_{ij} - \tilde{I}^l_{ij} \right\|. \tag{4.2}$$

Here, we use a simplified SSIM with a 3×3 block filter instead of a Gaussian, and set $\alpha = 0.85$.

Disparity Smoothness Loss We encourage disparities to be locally smooth with an $L1$ penalty on the disparity gradients ∂d. As depth discontinuities often occur at image gradients, similar to [91], we weight this cost with an edge-aware term using the image gradients ∂I,

$$C^l_{ds} = \frac{1}{N}\sum_{i,j} \left|\partial_x d^l_{ij}\right| e^{-\|\partial_x I^l_{ij}\|} + \left|\partial_y d^l_{ij}\right| e^{-\|\partial_y I^l_{ij}\|}. \tag{4.3}$$

Left-Right Disparity Consistency Loss To produce more accurate disparity maps, we train our network to predict both the left and right image disparities, while only being given the left view as input to the convolutional part of the network. To ensure coherence, we introduce an $L1$ left-right disparity consistency penalty as part of our model. This cost attempts to make the left-view disparity map be equal to the *projected* right-view disparity map,

$$C^l_{lr} = \frac{1}{N}\sum_{i,j} \left| d^l_{ij} - d^r_{ij+d^l_{ij}} \right|. \tag{4.4}$$

Like all the other terms, this cost is mirrored for the right-view disparity map and is evaluated at all of the output scales.

At test time, our network predicts the disparity at the finest scale level for the left image d^l, which has the same resolution as the input image. Using the known

camera baseline and focal length from the training set, we then convert from the disparity map to a depth map. While we also estimate the right disparity d^r during training, it is not used at test time.

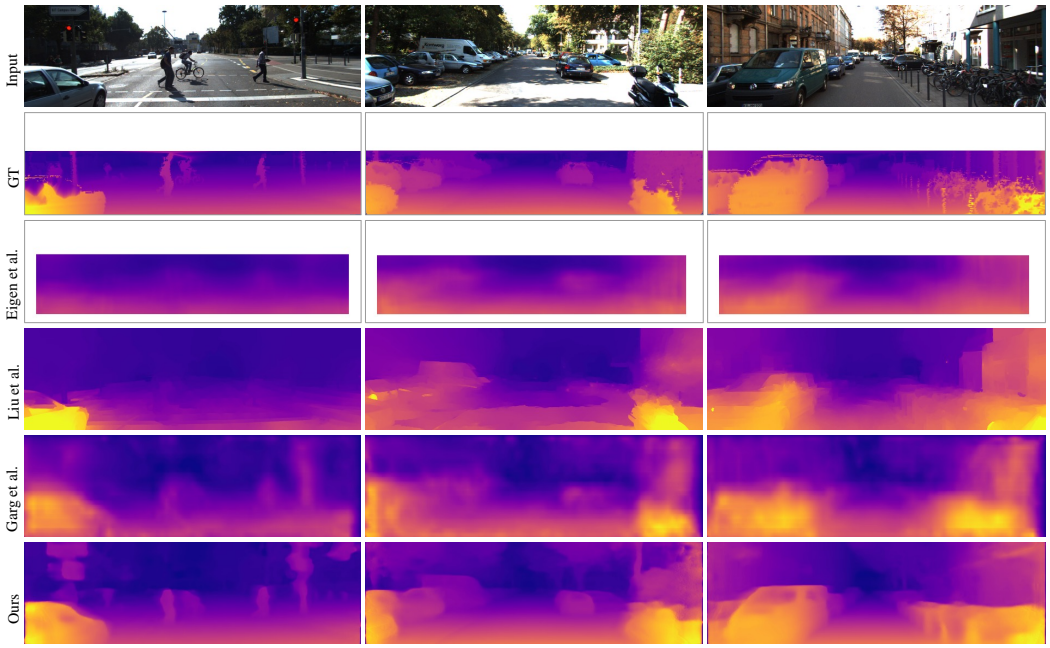


Figure 4.4: Qualitative results on the KITTI Eigen Split. The ground truth velodyne depth being very sparse, we interpolate it for visualization purposes. Our method does better at resolving small objects such as the pedestrians and poles.

4.2 Implementation and Results

Here we compare the performance of our approach to both supervised and unsupervised single view depth estimation methods. We train on rectified stereo image pairs, and do not require any supervision in the form of ground truth depth. Existing single image datasets, such as [92, 52], that lack stereo pairs, are not suitable for evaluation. Instead we evaluate our approach using the popular KITTI 2015 [86] dataset. To evaluate our image formation model, we compare to a variant of our algorithm that uses the original Deep3D [64] image formation model and a modified one, Deep3Ds, with an added smoothness constraint. We also evaluate our approach with and without the left-right consistency constraint.

4.2.1 Implementation Details

The network which is implemented in TensorFlow [93] contains 31 million trainable parameters, and takes on the order of 25 hours to train using a single Titan X GPU on a dataset of 30 thousand images for 50 epochs. Inference is fast and takes less than 35 ms, or more than 28 frames per second, for a 512×256 image, including transfer times to and from the GPU. Please see our code[1] for more details.

During optimization, we set the weighting of the different loss components to $\alpha_{ap} = 1$ and $\alpha_{lr} = 1$. The possible output disparities are constrained to be between 0 and d_{max} using a scaled sigmoid non-linearity, where $d_{max} = 0.3\times$ the image width at a given output scale. As a result of our multi-scale output, the typical disparity of neighboring pixels will differ by a factor of two between each scale (as we are upsampling the output by a factor of two). To correct for this, we scale the disparity smoothness term α_{ds} with r for each scale to get equivalent smoothing at each level. Thus $\alpha_{ds} = 0.1/r$, where r is the downscaling factor of the corresponding layer with respect to the resolution of the input image that is passed into the network.

For the non-linearities in the network, we used exponential linear units [94] instead of the commonly used rectified liner units (ReLU) [95]. We found that ReLUs tended to prematurely fix the predicted disparities at intermediate scales to a single value, making subsequent improvement difficult. Following [96], we replaced the usual deconvolutions with a nearest neighbor upsampling followed by a convolutions. We trained our model from scratch for 50 epochs, with a batch size of 8 using Adam [97], where $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$. We used an initial learning rate of $\lambda = 10^{-4}$ which we kept constant for the first 30 epochs before halving it every 10 epochs until the end. We initially experimented with progressive update schedules, as in [48], where lower resolution image scales were optimized first. However, we found that optimizing all four scales at once led to more stable convergence. Similarly, we use an identical weighting for the loss of each scale as we found that weighting them differently led to unstable convergence. We experimented with batch normalization [98], but found that it did not produce a

[1] Available at https://github.com/mrharicot/monodepth

significant improvement, and ultimately excluded it.

Data augmentation is performed on the fly. We flip the input images horizontally with a 50% chance, taking care to also swap both images so they are in the correct position relative to each other. We also added color augmentations, with a 50% chance, where we performed random gamma, brightness, and color shifts by sampling from uniform distributions in the ranges $[0.8, 1.2]$ for gamma, $[0.5, 2.0]$ for brightness, and $[0.8, 1.2]$ for each color channel separately.

Resnet50 For the sake of completeness, and similar to [61], we also show a variant of our model using Resnet50 [99] as the encoder, the rest of the architecture, parameters and training procedure staying identical. This variant contains 48 million trainable parameters and is indicated by **resnet** in result tables.

Post-processing In order to reduce the effect of stereo disocclusions which create disparity ramps on both the left side of the image and of the occluders, a final post-processing step is performed on the output. For an input image I at test time, we also compute the disparity map d_l' for its horizontally flipped image I'. By flipping back this disparity map we obtain a disparity map d_l'', which aligns with d_l but where the disparity ramps are located on the right of occluders as well as on the right side of the image. We combine both disparity maps to form the final result by assigning the first 5% on the left of the image using d_l'' and the last 5% on the right to the disparities from d_l. The central part of the final disparity map is the average of d_l and d_l'. This final post-processing step leads to both better accuracy and less visual artifacts at the expense of doubling the amount of test time computation. We indicate such results using **pp** in result tables.

4.2.2 KITTI

We present results for the KITTI dataset [86] using two different test splits, to enable comparison to existing works. In its raw form, the dataset contains $42,382$ rectified stereo pairs from 61 scenes, with a typical image being 1242×375 pixels in size.

KITTI Split First we compare different variants of our method including different image formation models and different training sets. We evaluate on the 200 high quality disparity images provided as part of the official KITTI training set, which

Method	Supervised	Dataset	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Train set mean	No	K	0.361	4.826	8.102	0.377	0.638	0.804	0.894
Eigen et al. [57] Coarse °	Yes	K	0.214	1.605	6.563	0.292	0.673	0.884	0.957
Eigen et al. [57] Fine °	Yes	K	0.203	1.548	6.307	0.282	0.702	0.890	0.958
Liu et al. [54] DCNF-FCSP FT *	Yes	K	0.201	1.584	6.471	0.273	0.68	0.898	0.967
Ours No LR	No	K	0.152	1.528	6.098	0.252	0.801	0.922	0.963
Ours	No	K	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Ours	No	CS + K	0.124	1.076	5.311	0.219	0.847	0.942	0.973
Ours pp	No	CS + K	0.118	0.923	5.015	0.210	0.854	0.947	**0.976**
Ours resnet pp	No	CS + K	**0.114**	**0.898**	**4.935**	**0.206**	**0.861**	**0.949**	**0.976**
Garg et al. [65] L12 Aug 8× cap 50m	No	K	0.169	1.080	5.104	0.273	0.740	0.904	0.962
Ours cap 50m	No	K	0.140	0.976	4.471	0.232	0.818	0.931	0.969
Ours cap 50m	No	CS + K	0.117	0.762	3.972	0.206	0.860	0.948	0.976
Ours pp cap 50m	No	CS + K	0.112	0.680	3.810	0.198	0.866	0.953	**0.979**
Ours resnet pp cap 50m	No	CS + K	**0.108**	**0.657**	**3.729**	**0.194**	**0.873**	**0.954**	**0.979**
Our pp uncropped	No	CS + K	0.134	1.261	5.336	0.230	0.835	0.938	0.971
Ours resnet pp uncropped	No	CS + K	0.130	1.197	5.222	0.226	0.843	0.940	0.971
			Lower is better				Higher is better		

Table 4.3: Results on KITTI 2015 [86] using the split of Eigen et al. [57]. For training, K is the KITTI dataset [86] and CS is Cityscapes [87]. The predictions of Liu et al. [54]* are generated on a mix of the left and right images instead of just the left input images. For a fair comparison, we compute their results relative to the correct image. As in the provided source code, Eigen et al. [57]° results are computed relative to the velodyne instead of the camera. Garg et al. [65] results are taken directly from their paper. All results, except [57], use the crop from [65]. We also show our results with the same crop and maximum evaluation distance. The last two rows are computed on the uncropped ground truth.

covers a total of 28 scenes. The remaining 33 scenes contain 30,159 images from which we keep 29,000 for training and the rest for evaluation. While these disparity images are much better quality than the reprojected velodyne laser depth values, they have CAD models inserted in place of moving cars. These CAD models result in ambiguous disparity values on transparent surfaces such as car windows, and issues at object boundaries where the CAD models do not perfectly align with the images. In addition, the maximum depth present in the KITTI dataset is on the order of 80 meters, and we cap the maximum predictions of all networks to this value. Results are computed using the depth metrics from [57] along with the *D1-all* disparity error from KITTI [86]. The metrics from [57] measure error in both meters from the ground truth and the percentage of depths that are within some threshold from the correct value. It is important to note that measuring the error in depth space while the ground truth is given in disparities leads to precision issues. In particular, the non-thresholded measures can be sensitive to the large errors in depth caused by prediction errors at small disparity values.

In Table 4.2, we see that in addition to having poor scaling properties (in terms

of both resolution and the number of disparities it can represent), when trained from scratch with the same network architecture as ours, the Deep3D [64] image formation model performs poorly. From Fig. 4.6 we can see that Deep3D produces plausible image reconstructions but the output disparities are inferior to ours. Our loss outperforms both the Deep3D baselines and the addition of the left-right consistency check increases performance in all measures. In Fig. 4.5 we illustrate some zoomed in comparisons, clearly showing that the inclusion of the left-right check improves the visual quality of the results. Our results are further improved by first pre-training our model with additional training data from the Cityscapes dataset [87] containing 22,973 training stereo pairs captured in various cities across Germany. This dataset brings higher resolution, image quality, and variety compared to KITTI, while having a similar setting. We cropped the input images to only keep the top 80% of the image, removing the very reflective car hoods from the input. Interestingly, our model trained on Cityscapes alone does not perform very well numerically. This is likely due to the difference in camera calibration between the two datasets, but there is a clear advantage to fine-tuning on data that is related to the test set.

Eigen Split To be able to compare to existing work, we also use the test split of 697 images as proposed by [57] which covers a total of 29 scenes. The remaining 32 scenes contain 23,488 images from which we keep 22,600 for training and the rest for evaluation, similarly to [65]. To generate the ground truth depth images,

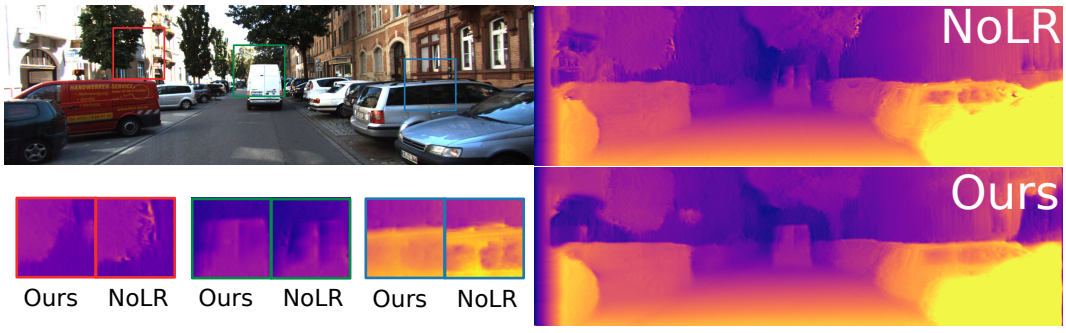


Figure 4.5: Comparison between our method with and without the left-right consistency. Our consistency term produces superior results on the object boundaries. Both results are shown without post-processing.

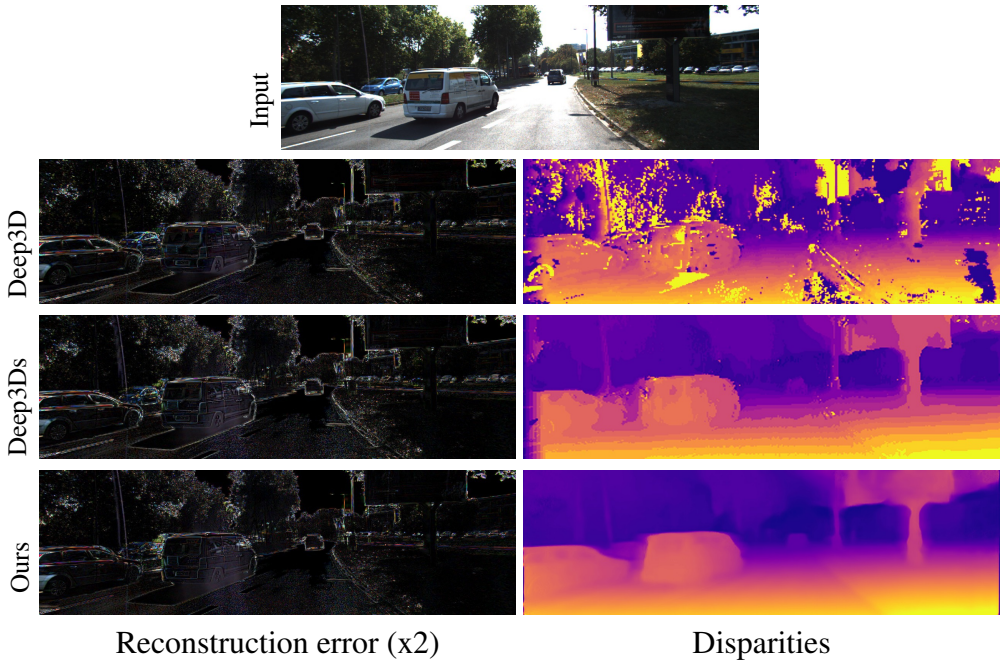


Figure 4.6: Image reconstruction error on KITTI. While all methods output plausible right views, the Deep3D image formation model without smoothness constraints does not produce valid disparities.

we reproject the 3D points viewed from the velodyne laser into the left input color camera. Aside from only producing depth values for less than 5% of the pixels in the input image, errors are also introduced because of the rotation of the Velodyne, the motion of the vehicle and surrounding objects, and also incorrect depth readings due to occlusion at object boundaries. To be fair to all methods, we use the same crop as [57] and evaluate at the input image resolution. With the exception of Garg et al.'s [65] results, the results of the baseline methods are recomputed by us given the authors's original predictions to ensure that all the scores are directly comparable. This produces slightly different numbers than the previously published ones, *e.g.* in the case of [57], their predictions were evaluated on much smaller depth images ($1/4$ the original size). For all baseline methods we use bilinear interpolation to resize the predictions to the correct input image size.

Table 5.1 shows quantitative results with some example outputs shown in Fig. 4.4. We see that our algorithm outperforms all other existing methods, including those that are trained with ground truth depth data. We again see that pre-training on the Cityscapes dataset improves the results over using KITTI alone.

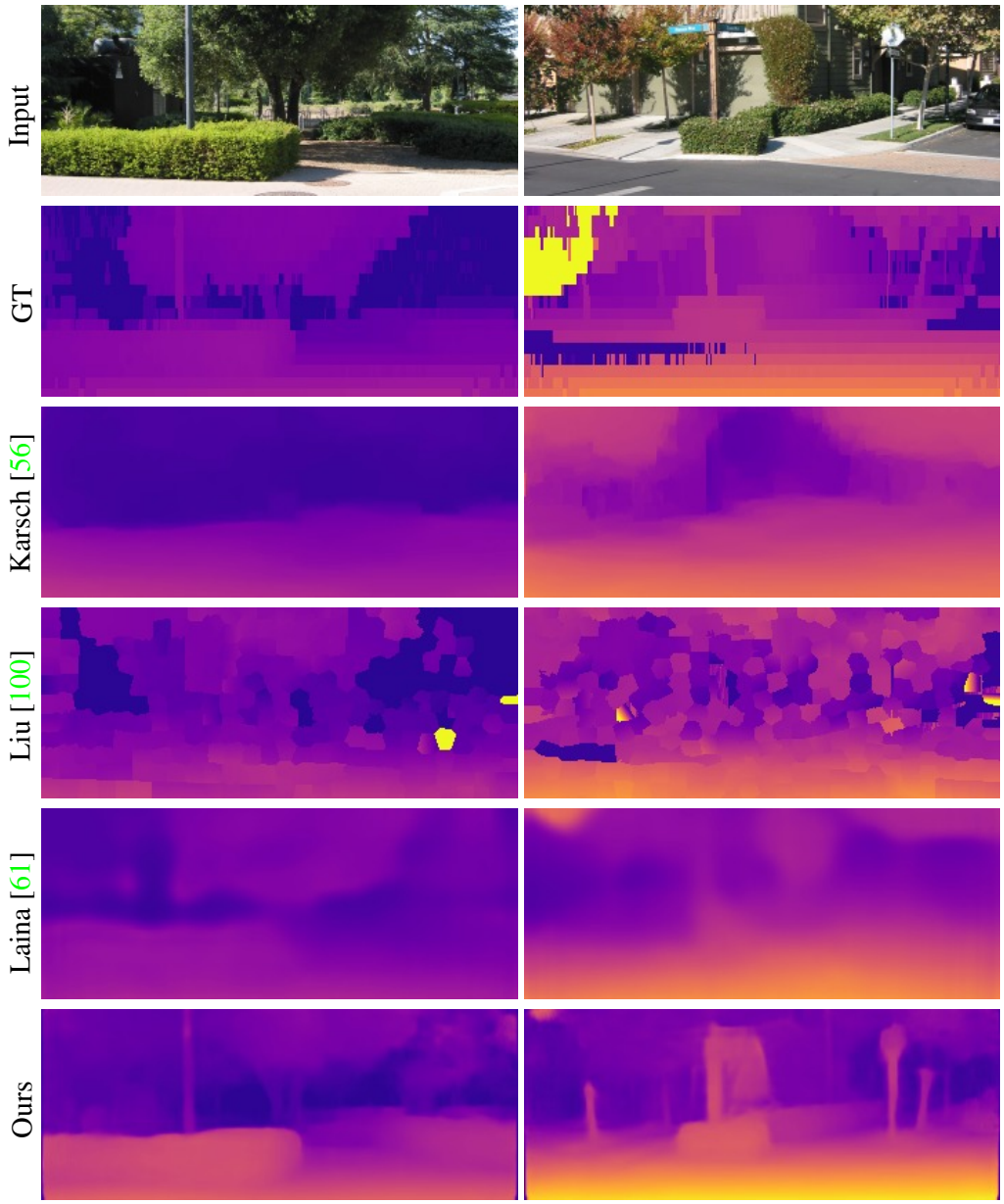


Figure 4.7: Our method achieves superior qualitative results on Make3D despite being trained on a different dataset (Cityscapes).

4.2.3 Stereo

We also implemented a stereo version of our model, see Fig. 4.8, where the network's input is the concatenation of both left and right views. Perhaps unsurprisingly, the stereo models outperforms our monocular network on every single metric, especially on the *D1-all* disparity measure, as can be seen in Table 4.2. This model was only trained for 12 epochs as it becomes unstable if trained for longer.

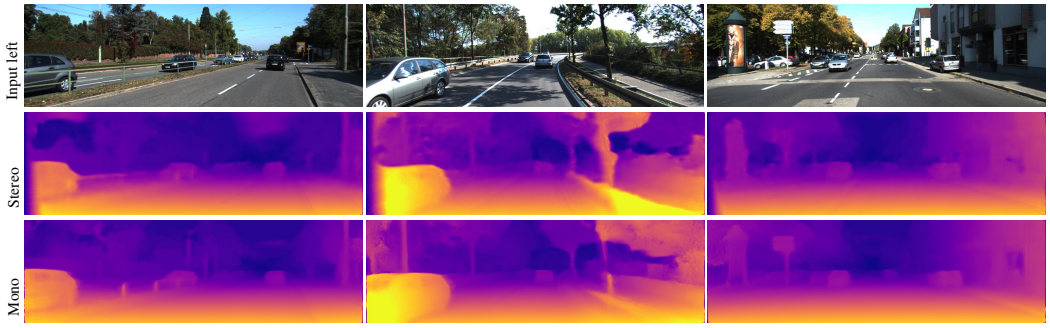


Figure 4.8: Our stereo results. While the stereo disparity maps contains more detail, our monocular results are comparable.

4.2.4 Make3D

To illustrate that our method can generalize to other datasets, here we compare to several fully supervised methods on the Make3D test set of [52]. Make3D consists of only RGB/Depth pairs and no stereo images, thus our method cannot train on this data. We use our network trained only on the Cityscapes dataset and despite the dissimilarities in the datasets, both in content and camera parameters, we still achieve reasonable results, even beating [56] on one metric and [100] on three. Due to the different aspect ratio of the Make3D dataset we evaluate on a central crop of the images. In Table 4.4, we compare our output to the similarly cropped results of the other methods. As in the case of the KITTI dataset, these results would likely be improved with more relevant training data. A qualitative comparison to some of the related methods is shown in Fig. 4.7. While our numerical results are not as good as the baselines, qualitatively, we compare favorably to the supervised competition.

Method	Sq Rel	Abs Rel	RMSE	$\log_{10}$
Train set mean*	15.517	0.893	11.542	0.223
Karsch et al. [56]*	4.894	0.417	8.172	0.144
Liu et al. [100]*	6.625	0.462	9.972	0.161
Laina et al. [61] berHu*	**1.665**	**0.198**	**5.461**	**0.082**
Ours with Deep3D [64]	17.18	1.000	19.11	2.527
Ours	11.990	0.535	11.513	0.156
Ours pp	7.112	0.443	8.860	0.142

Table 4.4: Results on the Make3D dataset [52]. All methods marked with an * are supervised and use ground truth depth data from the Make3D training set. Using the standard C1 metric, errors are only computed where depth is less than 70 meters in a central image crop.

4.2.5 Generalizing to Other Datasets

Finally, we illustrate some further examples of our model generalizing to other datasets in Figure 4.9. Using the model only trained on Cityscapes [87], we tested on the CamVid driving dataset [101]. We also captured a 60,000 frame dataset, at 10 frames per second, taken in an urban environment with a wide angle consumer 1080p stereo camera. Fine-tuning the Cityscapes pre-trained model on this dataset produces visually convincing depth images for a test set that was captured with the same camera on a different day, see Figure 4.9.

4.2.6 Limitations

Even though both our left-right consistency check and post-processing improve the quality of the results, there are still some artifacts visible at occlusion boundaries due to the pixels in the occlusion region not being visible in both images. Explicitly reasoning about occlusion during training [102, 103] could improve these issues. It is worth noting that depending how large the baseline between the camera and the depth sensor, fully supervised approaches also do not always have valid depth for all pixels.

Our method requires rectified and temporally aligned stereo pairs during training, which means that it is currently not possible to use existing single-view datasets for training purposes *e.g.* [92]. However, it is possible to fine-tune our model on application specific ground truth depth data.

Finally, our method mainly relies on the image reconstruction term, meaning that specular [104] and transparent surfaces will produce inconsistent depths. This could be improved with more sophisticated similarity measures [45].

4.3 Conclusion

In this chapter, we presented a self-supervised deep neural network model for monocular depth estimation. Instead of using aligned ground truth depth data, which is both rare and costly, we exploit the ease with which binocular stereo data can be captured and reframe the depth learning task as an indirect one. We do so by predicting the appearance of the scene from the viewpoint of the second image in the stereo pair, via a differentiable depth-based image formation model. Our novel loss function enforces consistency between the predicted depth maps from each camera view during training, improving predictions. Our results are superior to published fully supervised baselines, which is encouraging for future research

Figure 4.9: Qualitative results on Cityscapes, CamVid, and our own urban dataset captured on foot. For more results please see our video.

that does not require expensive ground truth depth. We have also shown that our model can generalise to unseen datasets and still produce visually plausible depth maps.

In the next Chapter we will look at improving the generalization of the method we introduced here by allowing it to train on any monocular sequence.

Chapter 5

Digging into Self-Supervised Monocular Depth Estimation

As in Chapter 4, we seek to automatically infer a dense depth map from a single input colour image. We extend the previously covered model, by allowing it to train using frames from a monocular video, or temporal sequences, instead of stereo pairs at training time, thus reducing the amount of supervision.

As mentionned in Chapter 4, one way to train deep depth estimation models is to use **ground truth depth images** paired with their corresponding intensity image as a supervision signal, *e.g.* [52, 57]. However, collecting large and varied training datasets with ground truth depth is itself a formidable challenge. Recently, several approaches have shown that it is instead possible to train monocular depth estimation models using only synchronized **stereo pairs** at training time [64, 65, 105]. While easier than laser-scanning, this still requires the collection of binocular stereo images. As an alternative, [106] successfully showed that **monocular video** sequences can be used for training, but with a drop in the quality of test time depth predictions.

Among the two self-supervised approaches, monocular video is an attractive alternative to stereo based supervision, but it introduces its own set of challenges. In addition to estimating depth, training of the former means also estimating the egomotion between temporal image pairs. So far, that meant training a *separate* pose network that takes a sequence of frames as input, and outputs the correspond-

ing camera transformations. Using stereo data for training makes the camera-pose estimation a one-time offline step, but can cause issues related to occlusion and texture-copy artifacts [105]. To address these issues, we propose a new architecture that shares weights between the pose and depth estimation networks in the monocular setting, and also drastically reduces texture-copy artifacts by performing image sampling at the input scale (see Fig. 5.1).

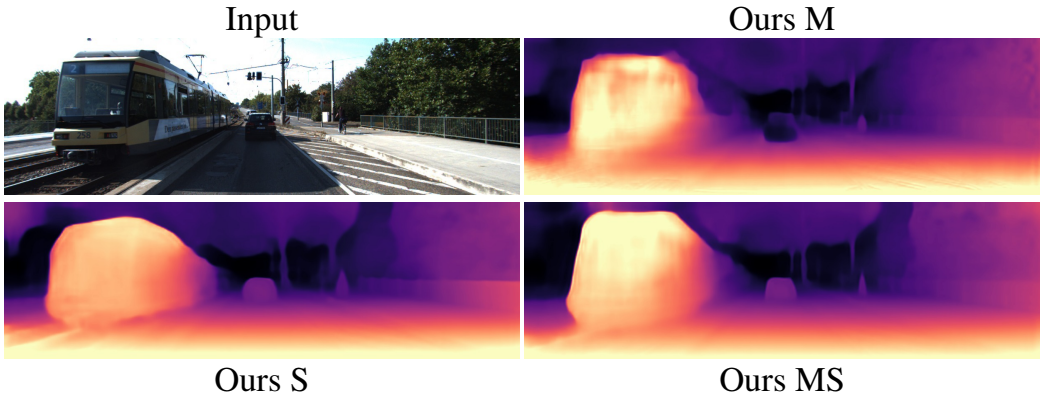


Figure 5.1: Estimated depth. Our method significantly outperforms all previously published self-supervised depth estimation methods, whether training with monocular supervision (M), stereo supervision (S), or both (MS).

In this chapter, we propose several architectural and loss innovations that, when combined, lead to large improvements in monocular depth estimation, using either monocular video, stereo pairs, or a combination as training data. We present the following three contributions:

- A novel architecture and loss function for the problem of self-supervised monocular depth estimation. Our models achieve state-of-the-art results on the KITTI dataset [86] in the self-supervised setting, and simplify many of the common components found in existing networks that use either monocular or stereo training data.

- A detailed evaluation, and list of best practices, that highlights how each component of our model lends itself to improved test time performance.

- Finally, we show that KITTI, the most commonly used evaluation dataset for monocular depth estimation, is not an informative benchmark for real-world

monocular training moving forward, as it contains only a limited number of independently moving objects.

The approach presented in Chapter 4 was published as [105] and presented at CVPR in 2017. It has since then been extended by training with semi-supervised data [107, 108] and by including additional temporal information [109, 110].

Using rectified stereo data at training time has the benefit of reducing the depth estimation problem to that of a 1D disparity search problem. Importantly, when the stereo cameras are synchronised they are not affected by scene motion. However, occlusion and dis-occlusion are still an issue as parts of the scene may not be visible given a fixed camera baseline. Another challenge is that large amounts of stereo data are not as readily available as traditional monocular videos. In this chapter we show that with careful design choices, training with only a single view can reach the performance of stereo training on existing datasets and results can be improved even further by including temporal information.

Self-supervised Monocular Training

A less constrained form of self-supervision is to use only monocular videos during training. Here, a similar type of image reconstruction loss as in the stereo supervision case is used. However, instead of using stereo pairs, neighbouring temporal frames from the input video provide the training signal. The added challenge during training is that the depth estimation model also has to estimate the camera pose between the input frames, in addition to coping with non-rigid scene motion. Unlike the stereo training regime that can cope with static scenes, in the monocular training setting there must be some non-trivial camera motion between the two image pairs or else there will be no training signal.

In one of the first works that used monocular self-supervision, [106] trained a depth estimation network along with a separate pose estimation network. The goal of the pose network is to predict the relative camera transformation between each subsequent temporal image pair. This estimated camera pose is only needed at training time to help constrain the depth estimation network. To deal with non-rigid scene motion, an additional motion explanation mask was learned, allowing

the model to ignore specific regions that violated the rigid scene assumption when computing the image reconstruction loss during training. However, later iterations of their model disables this term and achieves superior performance[1]. Inspired by [111], [112] proposed a more sophisticated motion model, where each image pixel is explained by a combination of multiple rigid transformations defined by K different motion masks. However, this motion decomposition is not fully evaluated making it difficult to know its utility. [113] also decompose motion into rigid and non-rigid components using the projected depth and a predicted residual optical flow to explain object motion. This improves the final optical flow estimation, but the authors report no improvement when jointly training the residual flow estimator and the depth prediction network. Thus, their depth estimation is not improved by this additional non-rigid motion term.

Recent approaches are beginning to close the performance gap between monocular and stereo based training supervision by making use of several different constraints during training. [114] constrained the output depth to be consistent with predicted surface normals. [115] used an approximate ICP based geometry matching loss to enforce temporal depth consistency. [116] observed that the commonly used depth smoothness term has a preference towards smaller depth maps, making the training of these models more unstable. To overcome this limitation, they normalised the predicted depth maps before computing the smoothness term, resulting in better performance. However, most of these methods do not explicitly deal with scene motion and, as a result, fail on moving objects during training.

In this chapter we propose several architectural and loss changes that simplify many of these recent approaches and which result in superior performance. We show that object motion is a challenge for monocular based methods and can be alleviated by either ignoring those scenes during training or by the inclusion of stereo data at training time when available.

[1] https://github.com/tinghuiz/SfMLearner

5.1 Method

Here we describe our monocular depth prediction network, which requires only a single color image at test time. We first review the key idea behind self-supervised training for monocular depth estimation, and then describe our depth and pose estimation networks and combined training loss.

5.1.1 Self-supervised Training

The key idea behind self-supervised depth estimation is to frame the learning problem as one of novel view-synthesis, essentially teaching the network to predict the appearance of a target image from the image taken from *another* viewpoint. By constraining the network to perform image synthesis using an intermediary variable, in our case depth or disparity, we can then extract out this interpretable depth from the model. This is an ill-posed problem as there are an extremely large number of possible wrong depths per pixel which can correctly reconstruct the novel view given a relative transformation between those two views. This is essentially the same problem faced by binocular and multi-view stereo methods. These methods typically address this ambiguity through enforcing smoothness in the depth maps by computing photo-consistency on patches and solving for the optimal depths for each pixel in a global optimization framework [41].

Like [65, 105, 106], we also formulate our problem as the minimization of a photometric reprojection error. For a target color image I_t, we predict a dense depth map D_t such that, given source views $I_{t'}$ and the relative rigid transformation between those views and the target view $T_{t\to t'}$ which minimizes the photometric reprojection error L_p,

$$\arg\min_{D_t} L_p, \tag{5.1}$$

$$\text{with} \quad L_p = \sum_{t'} pe(I_t, I_{t'\to t}), \tag{5.2}$$

$$\text{and} \quad I_{t'\to t} = I_{t'}\Big[proj(D_t, T_{t\to t'}, K)\Big]. \tag{5.3}$$

Here pe is a photometric reconstruction error, *e.g.* the L1 or L2 distance in pixel

space, $proj$ are the resulting 2D coordinates of the projected depths D_t into $I_{t'}$ and $\left\langle \right\rangle$ is the sampling operator. For simplicity of notation we assume the intrinsics K of all the views to be identical, they can however be different. Following [88, 105] we use bilinear sampling to sample the source images, which is *locally* sub-differentiable. This locality is a limitation which we overcome by making use of a multi-scale reconstruction approach, further improved by our *upsampled* multi-scale sampling.

Until now, we have assumed that we know the relative transformations $T_{t\to t'}$ between our target view I_t and source views $I_{t'}$. This is generally not the case for monocular sequences. However, if the target image and its source image are from a rectified stereo pair, the transformation between the pairs is purely horizontal. Stereo based training approaches like [65, 105] make use of this constraint when training single frame depth estimation models. For more general monocular training, [106] showed that it is possible to train a second pose estimation network jointly with the depth estimation network. Where the only goal is to predict the relative poses $T_{t\to t'}$ used in the projection function $proj$. Having to also solve for the camera transformations, in addition to the depth, our objective becomes

$$\arg\min_{D_t, T_{t\to t'}} L_p. \tag{5.4}$$

5.1.2 Improved Monocular Depth Estimation

Here, we describe several improvements to existing self-supervised depth estimation models. We step through the details of our approach, working through the design decisions taken on pose, loss functions, and scale.

Pose Estimation

The majority of current state-of-the-art models for monocular depth estimation with monocular training data employ a very similar architecture *e.g.* [106, 115, 113, 116]. This involves a standard U-Net model [117] for the depth estimator and a separate pose estimation network, see Fig. 5.2. The pose estimation network, which is not necessary for depth estimation at test time, takes as input a sequence of two or more

concatenated input frames and estimates the pose transformation between them. We argue that this base design is sub-optimal. Concatenating several input frames only makes learning harder as the training set is still the same size but the dimensionality of the input data grows with the number of frames in the input sequence. Moreover, the pose estimation model has to learn the difficult task of structure-from-motion from a short ordered sequence, with the only supervision signal being from the reprojection error. Improvements have been proposed by [116] who use direct methods in combination with the estimated depth maps, and [115] who use a 3d alignment loss between the predicted depth maps in the input sequence to improve both the pose and depth estimation. However, both these approaches are at the expense of a more complicated training procedure with only small test time improvements. These approaches build on the idea that the combination of the predicted monocular depth maps in the training sequence is a very strong geometric signal that can be used to better estimate the relative poses.

We instead propose a simple modification to the base architecture, that results in a significant improvement in depth accuracy as well as a reduction in the number of parameters that need to be learned. We make the observation that the deepest features of our depth encoder are only a small number of convolutional layers away from producing depth. We thus concatenate the last features from our depth encoder, which we then feed through a small three layer fully convolutional network *i.e.* the pose decoder, followed by a global average pooling, see Fig. 5.2. In effect, we are replacing the concatenation of the input images by the concatenation of the depth features. Our pose decoder is identical to the last three layers of the standard pose network from [106], but produces significantly better results. We essentially feed our small pose decoder deep abstract features which have an intrinsic understanding of the geometry of each of the input images, thus greatly facilitating its work.

Appearance Matching Loss

When computing the reprojection error from multiple source images, previous self-supervised depth estimation methods use the average reprojection error described

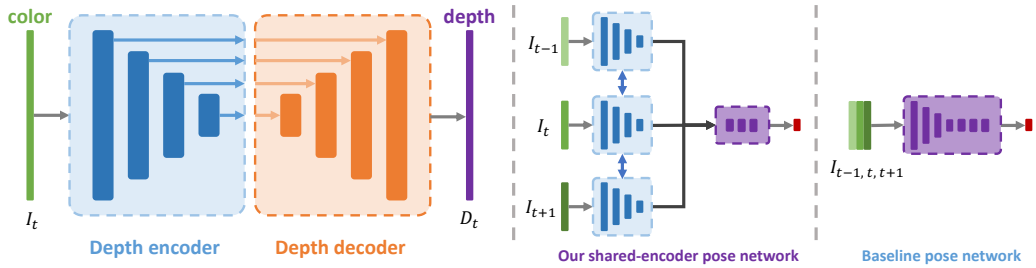


Figure 5.2: Overview of our network architecture. Unlike existing approaches to monocular depth estimation that make use of a separate pose network (right), we share weights between our depth encoder (left) and pose network (middle), resulting in faster convergence and improved performance. The pose estimation network outputs (red block) a 6-dimensional vector for each source image, representing the relative rigid transformation of the camera pose to the target frame.

in Eqn. (5.4). This causes problems with pixels which are visible in the target image but are *not visible* in some of the source images. If the network predicts the correct depth for such a pixel the corresponding colour in an occluded source image will likely not match the target one, inducing a photometric error penalty for being correct. Such problematic pixels are from two categories: out-of-view pixels due to ego-motion and dis-occluded pixels. As we show in Fig. 5.5, using an average reprojection error typically results in black holes (infinite depth) around image edges and soft occlusion boundaries, for each category respectively. The effect of out-of-view pixels can be reduced by simply ignoring such pixels in the reprojection loss [115, 112], this however doesn't handle dis-occluded pixels.

We propose an improvement which deals with both issues at once. Instead of averaging the photometric error per pixel over all source images, we simply use the per-pixel minimum. As we show in Fig. 5.5, this significantly reduces artifacts at image borders, improves the sharpness of occlusion boundaries, and leads to better accuracy.Following [89, 105], we use a combination of L1 and SSIM [90] as our photometric error function pe. Our final photometric loss is

$$L_p = \min_{t'} pe(I_t, I_{t'\to t}), \tag{5.5}$$

$$\text{where} \quad pe(I_a, I_b) = \alpha \frac{1 - \text{SSIM}(I_a, I_b)}{2} + (1-\alpha)\|I_a - I_b\|_1. \tag{5.6}$$

We also make use of edge aware smoothness on the predicted disparities

$$L_s = |\partial_x d_t^*| e^{-|\partial_x I_t|} + |\partial_y d_t^*| e^{-|\partial_y I_t|}, \tag{5.7}$$

$$\text{with} \quad d_t^* = d_t / \overline{d_t}, \tag{5.8}$$

where d_t^* is the mean-normalized inverse depth for I_t as used by [116].

Multi-scale Estimation

Because of the gradient locality of the bilinear sampler, existing models use multi-scale depth prediction and image reconstruction to constrain the training objective, where the total loss is typically the average of the individual losses at each scale. The original formulations from [65] and [105] compute the photometric error on downsampled images, which we observe has the tendency to create 'holes' in large low-texture regions, such as roads or the sky, in the intermediate lower resolution depth maps. This can be explained by the lack of texture information in these images, thus effectively making the photometric error more ambiguous. This in turn complicates the task of the depth estimation network which is then free to predict an incorrect depth for a given pixel at the low resolution resulting in a low reprojection error at that scale, which in turn leads to a large photometric error at higher resolutions.

We propose an improvement to this multi-scale formulation. Instead of computing the photometric error on the ambiguous low-resolution images, we first upsample the lower resolution depth maps to the input image resolution and then we warp and compute the photometric error *pe* at this higher input resolution. This effectively constrains the depth maps from each resolution to work towards the exact same objective *i.e.* reconstructing the input high resolution target image as accurately as possible. We found that this significantly improves the depth accuracy, while also reducing the texture-copy artifacts which are very noticeable in the original multi-scale formulation as can be seen in Fig. 5.5. This is related to matching patches, a standard practice in stereo reconstruction [40], as a low-resolution disparity value will be responsible for reprojecting an entire patch of pixel in the high

resolution image.

Final Training Loss

We combine both our photometric and smoothness losses into a final training loss

$$L = L_p + \lambda L_s, \tag{5.9}$$

which is averaged per pixel, per scale, and per batch. During training, we set the weight of the smoothness term, λ, to 0.001.

5.2 Implementation and Results

In this section we compare the performance of our models to existing state-of-the-art on the KITTI [86] and Make3D [52] datasets.

5.2.1 Implementation Details

Our depth estimation network is based on the general U-Net architecture [117], which is essentially an encoder-decoder network, with skip connections enabling us to represent both deep abstract features as well as local information. We use a Resnet18 [99], pretrained on ImageNet [118], as our encoder. Our depth decoder is similar to [105] and uses ELU [119] activation functions except on the output depth layers which use Sigmoids. We then turn this output into depth by scaling and inverting the predicted disparities. We also make use of reflection padding, in place of zero padding, in the decoder layers, and return the value of the closest border pixels in the source image during reprojection, instead of zero, when samples land outside of the image boundaries. We found that these steps significantly improve the border artifacts commonly found in existing approaches *e.g.* [105].

We adopt the inverse depth normalization trick of [116] in all our experiments to avoid catastrophic shrinking of the estimated depth. For pose estimation, we follow [116] and predict the rotation using an axis-angle representation and scale the rotation and translation outputs by 0.01 and 0.001 respectively. When training with stereo data we use the left and right pairs as the input views, for monocular training we use a set of three frames, $t-1$, t, and $t+1$. All our networks were

implemented in PyTorch [120] and trained for 15 epochs, with a batch size of 8, using Adam [121], and with an initial learning rate of 10^{-4} for the first 10 epochs which was then dropped to 10^{-5}. Training on the KITTI dataset [86] with an input image of 128×416 pixels, which we refer to as Low Resolution (LR), takes 8 hours on a Titan X Maxwell, and twice as long for 192×640.

We also tried several other components which we found to not help performance. We experimented with using a feature reconstruction loss in the appearance matching term, similar to [122, 123, 110], by computing the L1 distance on the reprojected `relu_1` features from an ImageNet pretrained VGG16 [124] as our *pe* function but observed a slight decrease in accuracy compared to SSIM on KITTI. We explored using explanation masks [106], discrete motion models [112], and temporal depth consistency [112, 109], but none of which made any significant improvements in our implementation. Finally, we tried adding batch normalization [98] into the decoder but observed persistent ghosting artifacts in the predicted depth maps.

5.2.2 KITTI

We use the original data split from Eigen et al. [58] and follow Zhou's et al. [106] preprocessing to remove static frames and set the input sequence length to 3. We end up with 39810 and 4424 monocular triplets for training and validation. We used a single camera intrinsic matrix for all images, where we set the principal point of the camera to be centered and set the focal length as the average of all the focal lengths in KITTI. For stereo and mixed training (monocular and stereo) we fix the transformation between the two stereo frames to be a pure horizontal translation of fixed length. We perform horizontal flips and the following data augmentations during training with 50% chance: random brightness, contrast, saturation, and hue jittering with respective ranges of 0.2, 0.2, 0.2 and 0.1.

Results are presented using a cap at 80 meters. For our monocular models, we report results using the same median ground truth scaling as [106]. With stereo training data, scale can be inferred from the known camera baseline, and for fairness we do not use median scaling for our models that use any stereo supervision.

Method	Train	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Train set mean	D	0.361	4.826	8.102	0.377	0.638	0.804	0.894
Eigen [57]	D	0.203	1.548	6.307	0.282	0.702	0.890	0.890
Liu [54]	D	0.201	1.584	6.471	0.273	0.680	0.898	0.967
AdaDepth [125]	D*	0.167	1.257	5.578	0.237	0.771	0.922	0.971
Kuznietsov [107]	DS	0.113	0.741	4.621	0.189	0.862	0.960	**0.986**
SVSM [108]	D*S	0.102	0.700	4.681	0.200	0.872	0.954	0.978
SVSM FT [108]	DS	**0.094**	**0.626**	**4.252**	**0.177**	**0.891**	**0.965**	0.984
UnDeepVO [109]	MS	0.183	1.730	6.57	0.268	-	-	-
Zhan Temporal [110]	MS	0.144	1.391	5.869	0.241	0.803	0.928	0.969
Zhan FullNYU [110]	D*MS	0.135	1.132	5.585	0.229	0.820	0.933	0.971
Ours LR	MS	0.122	1.164	5.244	0.212	0.850	0.947	0.975
Ours	MS	**0.113**	**1.002**	**5.003**	**0.202**	**0.865**	**0.952**	**0.978**
Monodepth [105]	S	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Garg [65]†	S	0.152	1.226	5.849	0.246	0.784	0.921	0.967
Ours LR	S	0.118	1.044	5.264	0.216	0.849	0.944	0.974
Ours	S	**0.111**	**1.012**	**5.127**	**0.209**	**0.861**	**0.947**	**0.975**
Zhou [106]†	M	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang [114]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [115]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
GeoNet [113]	M	0.155	1.296	5.857	0.233	0.793	0.931	0.973
DDVO [116]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Ours LR	M	0.137	1.153	5.353	0.212	0.836	0.947	**0.978**
Ours	M	**0.133**	**1.111**	**5.182**	**0.209**	**0.845**	**0.950**	0.977

Table 5.1: Comparison to existing methods on KITTI 2015 [86] using the Eigen split. D refers to methods that use KITTI depth supervision at training time, D* use auxiliary depth supervision, S use stereo, and M use monocular supervision. †indicates newer results from the respective online implementations. LR is our model trained to predict at 128×416 resolution, otherwise we use 192×640.

In practice, we observe that this adds a small, but noticeable, improvement when included.

5.2.3 Results

We compare the results of several variants of our model trained with different types of self-supervision: monocular only, stereo only, and both. In Table 5.1 we see that we outperform all existing state-of-the-art approaches with the exception of models that make use of extensive depth supervision at training time *i.e.* [107, 108]. Our best performing variant uses both monocular and stereo training data where the results are most noticeable on metrics that are sensitive to large depth errors *i.e.* RMSE.

We can also see that our monocular supervised model, whether trained at 128×416 or 192×640, outperforms all previously published self-supervised methods, whether they used monocular supervision [106, 115, 116, 114], stereo supervision [65, 105] or both [109, 110].

To better understand how each component of our model contributes to the overall performance, in Table 5.2 we perform an ablation study by turning off different parts of our model, one at a time, in the monocular setting. 'PoseCNN' corresponds to our implementation of the strong baseline used in [116], with the standard separate pose encoder from [106], but without their direct visual odometry. We see that the inclusion of our shared pose encoder in 'Ours LR' improves the results. 'Avg. projection' is the average projection used by [106], as opposed to our minimum based projection from Eqn. 5.5. 'Low-res multi-scale' is the multi-scale reconstruction evaluation performed by [105], in contrast to our reconstruction which is performed at the input resolution. 'No pretraining' is our model that is not pretrained on ImageNet. As previously discussed in [115], we see that SSIM plays an important role in improving results. When combined in 'Ours LR', all these components lead to a significant performance improvement.

Monocular vs. Stereo Supervision

From Table 5.1, we see that monocular trained models perform worse than stereo based approaches. Moving objects are an issue for these approaches, but the KITTI dataset does not feature a large number of such objects. This problem commonly manifests itself as 'holes' in the predicted test time depth maps for objects that are typically observed moving during training *e.g.* the missing car ahead with all monocular methods in Fig. 5.4. Indeed, if the moving object has the same speed and direction as the camera, then the reprojection error is low if the disparity is 0 for that object. In KITTI, this can happen when the camera is following a moving car in the same lane at the same speed. (Unfortunately, monocular video alone is not sufficient to disambiguate pixels in the 'car following' scenario.)

Solely to explore this hypothesis, we trained another version of the 'Ours LR' model on a subset of the KITTI dataset, where we manually removed six entire sequences that featured a moving car in front of the main camera. This resulted in 37,294 training images. In Table 5.2, we see this model, denoted as 'No motion' performs better on the Sq Rel and RMSE metrics, despite having less training data. Further, we observe that pretraining on Cityscapes [87], which was shown to be very

Method	Train	Dataset	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
PoseCNN	M	K	0.150	1.221	5.639	0.222	0.805	0.941	0.976
Avg. reprojection	M	K	0.152	1.742	5.660	0.232	0.828	0.939	0.972
Low-res multi-scale	M	K	0.147	1.490	5.652	0.221	0.829	0.944	0.974
No pretraining	M	K	0.159	1.224	5.705	0.236	0.791	0.929	0.972
No SSIM	M	K	0.185	3.029	6.186	0.258	0.796	0.927	0.965
Ours LR	M	K	**0.133**	**1.158**	**5.370**	**0.208**	**0.841**	**0.949**	**0.978**
Ours LR	M	C	0.235	3.534	7.313	0.294	0.699	0.890	0.952
Ours LR	M	CK	0.141	1.431	5.597	0.219	0.838	0.946	0.975
Ours LR No motion	M	K*	0.138	1.072	5.231	0.212	0.835	0.948	0.978

Table 5.2: Results for different variants of our model that use monocular training on KITTI 2015 [86] using Eigen's split. For training, C is Cityscapes [87] and K is the KITTI [86]. All models are trained using a resolution of 128×416.

useful in the case of stereo training [105], actually hurts in our monocular setting. We hypothesize that the increased proportion of motion in Cityscapes "helps" the network learn unrealistic depths that actually reproject correctly. For example, cars moving at the same speed as the camera are mapped to a distance of infinity.

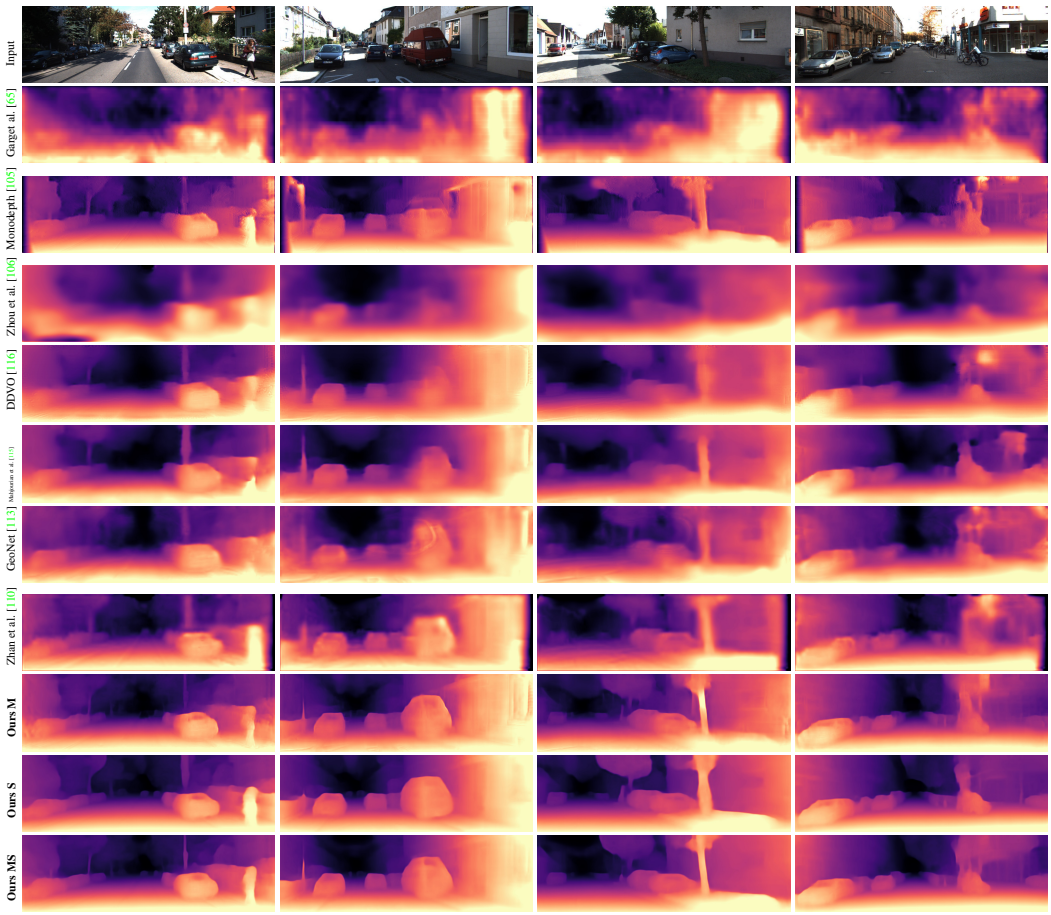


Figure 5.3: Qualitative results on the KITTI Eigen split. We can see that our approaches in the last three rows produce the sharpest depth maps, which is reflected by the quantitative results in Table 5.1.

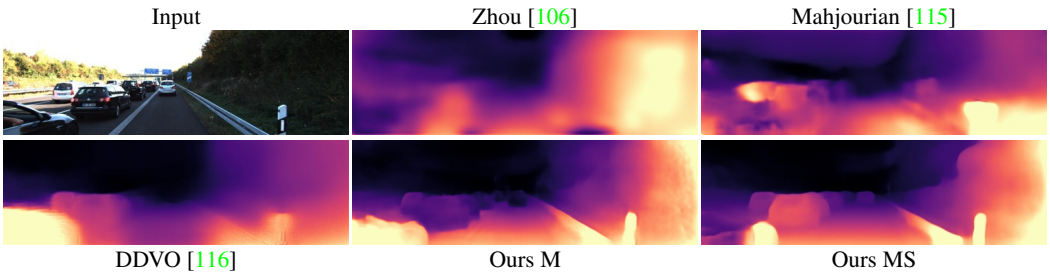


Figure 5.4: Monocular failures. Monocular based methods can sometimes fail at predicting depth for objects that were typically observed moving at training.

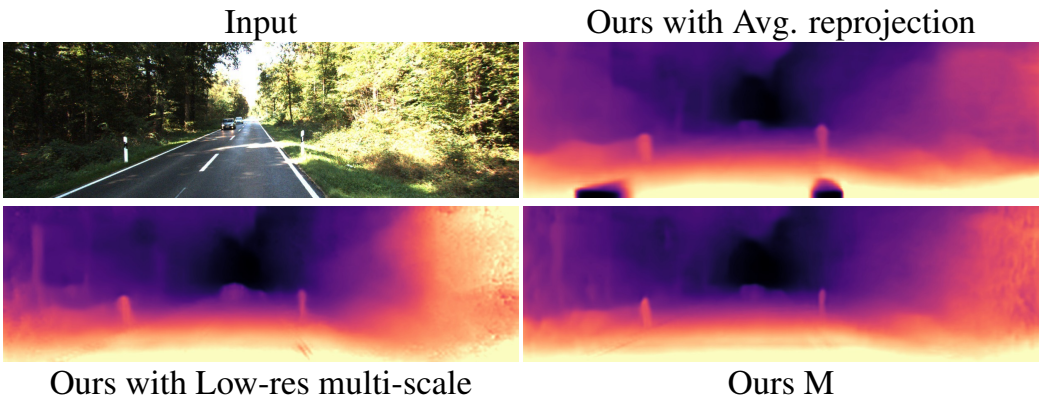


Figure 5.5: Comparison with 'Low-res multi-scale' and 'Avg. reprojection'. We see that our model results in less artifacts in the final depth image.

5.2.4 Additional Results

Here we present quantitative results on the Make3D dataset [52] using our model trained on KITTI monocular data. In Table 5.3 we outperform all methods that do not use depth supervision. However, caution should be taken with Make3D, as the ground truth depth and input images are not well aligned in the original dataset, causing potential evaluation issues. Qualitative results can be seen in Figs. 5.6 and

Method	Type	Abs Rel	Sq Rel	RMSE	$\log_{10}$
Train set mean	D	0.893	13.98	12.27	0.307
Karsch [56]	D	0.428	5.079	8.389	0.149
Liu [100]	D	0.475	6.562	10.05	0.165
Laina [61]	D	**0.204**	**1.840**	**5.683**	**0.084**
Monodepth [105]	S	0.544	10.94	11.760	0.193
Zhou [106]	M	0.383	5.321	10.470	0.478
DDVO [116]	M	0.387	4.720	8.090	0.204
Ours	M	**0.361**	**4.170**	**7.821**	**0.175**

Table 5.3: Make3D results. Our method outperforms all other self-supervised models, but falls short of [61] which was trained on this dataset using depth data.

5.7.

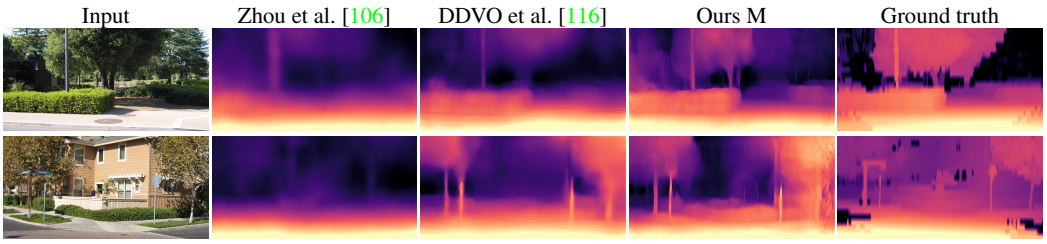


Figure 5.6: Qualitative results the Make3D dataset. All methods were trained on KITTI using monocular supervision.

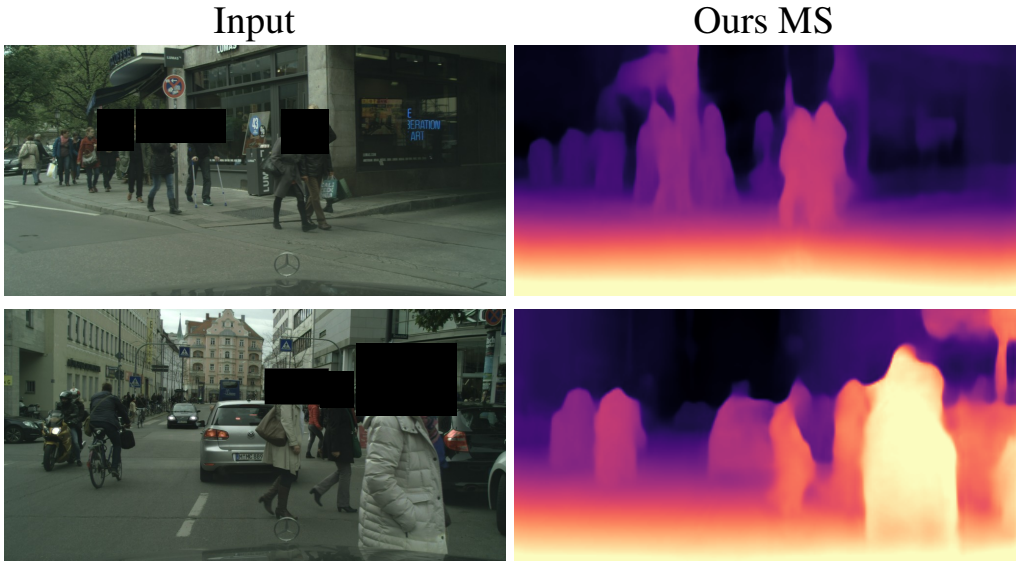


Figure 5.7: Qualitative results the Cityscapes dataset.

5.2.5 Convergence

In Fig. 5.9 we see that our model with the shared pose encoder converges faster, and to a better accuracy, compared to using a separate pose network. This test accuracy is not monitored during training as all our networks are trained for 15 epochs.

5.2.6 Effect of moving cars

Cars moving at the same speed as the camera, or close to it, can appear seemingly static for many consecutive frames - masquerading as if they were extremely far away objects. This often causes a monocular video based network to assign a very large depth to such cars, to reach a low reprojection error. These incorrect predictions can be seen at test time, as shown in Fig. 5.8. This problem is amplified by

pretraining on Cityscapes data [87], as this dataset contains many more sequences with such ambiguous situations, where the camera is following similar-speed cars. As we can see in Fig. 5.8, our monocular model, pretrained on Cityscapes, makes more dramatic mistakes in certain specific situations: cars just ahead are interpreted as "punched" out depths.

While we can mitigate the problem by excluding such nearly-matched-speed cars when training on KITTI, we found a better overall solution: we found that training with both monocular and stereo supervision addresses the issue directly.

5.2.7 Single scale test time evaluation

Our approach, like all self-supervised baselines, has no guarantee of producing results with a metric scale. Nonetheless, we anticipate that there could be value in estimating depth-outputs that are, without special measures, consistent with each other over the length of a video clip.

To evaluate the stability of our depth estimation, we modified the evaluation protocol from [106] to scale (or align) the predicted depths with a single scalar per method, instead of a *different* scalar per test depth map. In [106], the authors independently scale each predicted depth map by the ratio of the median of the ground truth and predicted depth map - for each individual test image. This is in contrast to stereo based training where the scale is known and as a result no additional scaling is required during the evaluation *e.g.* [65, 105]. This per-image depth scaling hides unstable scale estimation in both depth and pose estimation and presents a best case scenario for the monocular training case *i.e.* if a method outputs wildly varying scales for each sequence, then this evaluation protocol will significantly hide the issue. We thus modified the original protocol to instead use a single scale for all predicted depth maps of each method. For each method, we compute this single scale by averaging all the individual ratios of the depth medians on the *test* set. While this is still not ideal as it makes use of the ground truth depth, we believe it to be a more fair and representative of the performance of each method. We also calculated the standard deviation σ_{scale} of the individual scales, where lower values indicate more consistent output depth map scales. As

can be seen in Table 5.4, our method still outperforms all previously published self-supervised monocular methods, and is more stable.

Method	σ_{scale}	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou [106]†	0.201	0.278	2.636	7.428	0.335	0.576	0.836	0.930
Mahjourian [115]	0.184	0.234	1.874	6.616	0.297	0.642	0.871	0.948
GeoNet [113]	0.167	0.216	1.778	6.389	0.277	0.681	0.890	0.957
DDVO [116]	0.104	0.167	1.408	5.770	0.243	0.778	0.923	0.970
Ours LR	0.100	0.149	1.259	5.525	0.221	0.810	0.943	0.976
Ours	**0.089**	**0.139**	**1.165**	**5.253**	**0.214**	**0.832**	**0.946**	**0.977**

Table 5.4: Comparison to existing monocular supervised methods on KITTI 2015 [86] using the Eigen split using a *single* scale for each method. † indicates newer results from the respective online implementations. Here we compare our monocular trained model, where LR is our model trained to predict at 128×416 resolution, otherwise we use 192×640.

5.2.8 Odometry

In Table 5.5 we evaluate our pose estimation network following Zhou et al. [106] evaluation protocol. We trained our monocular model with 3 frames on sequences 0-8 from the KITTI odometry split and tested on 9-10. The absolute trajectory error (ATE) is averaged over all overlapping 5-frame snippets in the test sequences. Here, unlike [106] and others, that use specific architectures for the odometry task, we use the *same* architecture for this task as our depth estimation network with 3 temporal frames, and simply train it again from scratch on these new sequences. In order to compare our 3-frame model, we only use one relative transformation $T_{t \to t-1}$ and combine the frame-to-frame estimates to form local trajectories. For completeness we repeat the same process with Zhou's [106] predicted poses, which we indicate with a $*$ in the table. As we can see in Table 5.5, our frame-to-frame poses are better than both [106] and the previous state-of-the-art for monocular depth estimation [116]. They however fall short of the independent 5-frame estimations from previous self-supervised methods *i.e.* [115, 113].

5.2.9 Discussion

We determined that the following design decisions are important for realising better quality depth estimation models, across the self-supervised settings we studied here:

- Independent object motion is very challenging for monocular methods. This

	Sequence 09	Sequence 10
ORB-Slam [126]	0.014±0.008	0.012±0.011
Zhou [106]	0.021±0.017	0.020±0.015
Zhou [106]†	0.016±0.009	0.013±0.009
Mahjourian [115]	0.013±0.010	0.012±0.011
GeoNet [113]	0.012±0.007	0.012±0.009
DDVO [116]	0.045±0.108	0.033±0.074
Zhou* [106]	0.050±0.039	0.034±0.028
Ours LR	0.023±0.013	0.018±0.014

Table 5.5: Odometry results on the KITTI [86] odometry dataset. Results show the average absolute trajectory error, and standard deviation, in meters. †indicates newer results from the respective online implementations.

can be substantially mitigated by excluding frames with significant motion from training, if possible, or including them through the use of stereo pairs, where available. However it is clear that an explicit handling of object motion is required to truly exploit general monocular sequences.

- Higher resolution leads to increased performance. This is observed at both the final depth resolution, and during multi-scale image reconstruction.

- When predicting camera pose, it is beneficial to use shared weights between the depth and pose networks. This results in more stable training, fewer parameters to learn, and better performance.

- A minimum reprojection loss is a simple and elegant solution to deal with occluded pixels, compared to the typical average.

- Pre-training encoders on general image recognition tasks enables faster convergence, and results in better accuracy of depth estimation.

5.3 Conclusion

In this chapter, we presented a versatile model for self-supervised monocular depth estimation, building on the work outlined in Chapter 4. We showed that with some careful improvements on the network architecture and appearance prediction model,

our relatively simple model can outperform the existing self-supervised state-of-the-art depth estimation algorithms, whether they leverage self-supervision with monocular training data, stereo training data, or both. We have also observed that models trained with stereo images still outperform those that use only monocular videos, indicating that the pose estimation network can be improved upon.

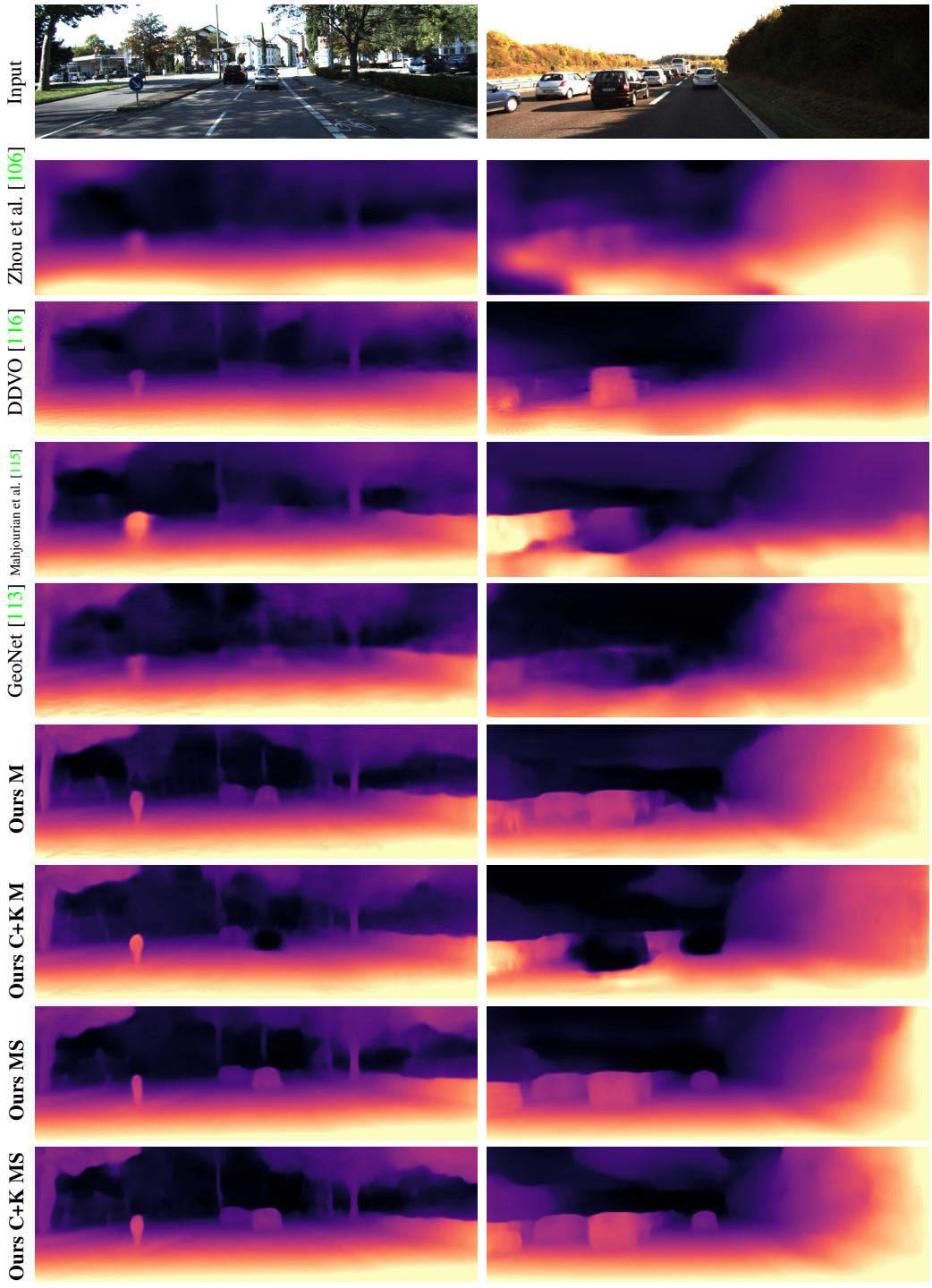


Figure 5.8: Moving cars and Cityscapes pretraining. All monocular methods tend to put cars which are ahead of the camera at a very large depth value. Pretraining on Cityscapes [87], which has more moving cars in the training data compared to KITTI, only makes things worse especially for our method. Training with the addition of stereo data significantly reduces the impact of this problem.

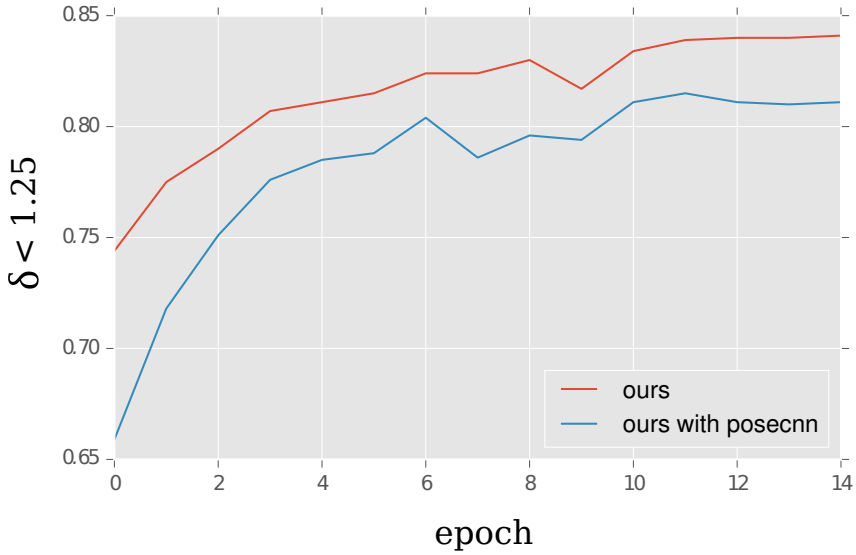


Figure 5.9: Convergence comparison. Here we compare the depth prediction accuracy of our model with our shared pose network (ours) and our model with the separate pose network (ours with posecnn) using the $\delta < 1.25$ metric on the KITTI test set, where higher values are better.

Chapter 6

Conclusion

In this thesis we explored passive 3D reconstruction techniques. Image-based reconstruction is a challenging task, as the mapping between 2D images to 3D surfaces is an under-constrained problem. Most binocular and multi-view reconstruction techniques essentially attempt to resolve the inherent ambiguity of this ill-posed problem. While passive methods based on appearance matching rely on establishing direct correspondences between input views, we decided to focus on challenging situations where it is not possible, namely multi-view specular surface reconstruction and single view depth estimation.

We hypothesised that passive reconstruction through appearance prediction is viable in situations where direct image correspondence is ill-defined. In this thesis we have given evidence that this is the case, and we outline our main results here.

We first focused, in Chapter 3, on the reconstruction of highly specular surfaces which is typically done with active methods. Reconstructing such surfaces is very difficult and traditional multi-view stereo methods relying on direct matching do not work. We showed that we could instead rely on indirect matching, by predicting the appearance of the reflections of the environment on the surface from multiple angles. By fusing multiple observations into a bayesian framework we were able to obtain detailed 3D models. Our method is the first to enable the reconstruction of highly specular objects in uncontrolled environments using photographs only.

In Chapter 4 we moved to a different situation which suffers from a severe ambiguity and consider the case of obtaining the depth of a single colour image.

Previous methods were built using machine learning and are typically trained with depth data which is difficult to obtain in large quantities. We instead built a convolutional neural network model capable of performing depth-based image rendering in a differentiable way. By using stereo images as training data, and leveraging our differentiable image formation model, we reframed the learning task as a novel-view synthesis one and learn depth by proxy as an indirect task. Inspired by traditional stereo methods we also introduced a novel left-right consistency loss to further improve the reconstruction accuracy. We showed a significant increase in sharpness and accuracy of the predicted depth maps on the KITTI dataset, outperforming all previously published methods, including the supervised ones.

Finally in Chapter 5 we extended the model presented in Chapter 4 to enable it to train from video sequences instead of stereo pairs. This problem is a more challenging one, as now a second neural network is also tasked with predicting the relative rigid transformation between the frames in the video. We make several key improvements, including modifying the pose estimation network to share its encoder with the depth estimation network. We also refine our appearance prediction model from the multiple nearby views to be more robust to low-texture regions and occlusions. The presented model significantly outperforms all self-supervised models on the KITTI dataset, even when using less source images at training time.

These three projects showed that an appearance prediction can be used in order to perform 3D reconstruction even when direct image correspondence isn't feasible.

Future work

While we have shown that passive reconstruction methods can achieve great results and shrink the accuracy gap versus active methods, the gap is still nonetheless present. There are however a lot more avenues left unexplored, and we outline some of them here.

Specular surface reconstruction The method we presented requires a manual creation of the silhouette of the object for each input image, which can be time consuming. It would be interesting to partially or fully automate the silhouette extraction

and improve the silhouette constraints in the surface optimisation. Given the impressive results obtained by convolutional neural networks, it would be interesting to train deep models to predict the normal and depth map of a single [127] input image, say from a synthetic dataset, or even to regress an entire volume from a multi-view dataset [128, 129].

Self-Supervised Monocular depth estimation While our current depth estimates are performed independently per frame, adding temporal consistency [56] would likely improve results, possibly through the use of a recurrent model [130]. It would also be interesting to investigate sparse input as an alternative training signal [131, 132]. Finally, while our model estimates per pixel depth, it would be interesting to also predict the full occupancy of the scene [133]. Existing datasets like KITTI slightly conceal motion-induced limitations, due to the relatively small number of moving objects at training time. We expect scene-flow based performance differences to become more apparent as the community moves to more complex and general-world training imagery. Finally, the network architecture presented in Chapter 5, opens the possibility of tacking multi-view stereo [134] in a self-supervised manner.

Bibliography

[1] Felipe Cucker. *Manifold mirrors: the crossing paths of the arts and mathematics*. Cambridge University Press, 2013. 10

[2] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003. 11

[3] Bela Julesz. Binocular depth perception and pattern recognition. *Information theory*, pages 212–224, 1961. 11

[4] Roger Y Tsai. Multiframe image point matching and 3-d surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):159–174, 1983. 11

[5] Masatoshi Okutomi and Takeo Kanade. A multiple-baseline stereo. *IEEE Transactions on pattern analysis and machine intelligence*, 15(4):353–363, 1993. 11

[6] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010. 12

[7] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 12

[8] Ivo Ihrke, Kiriakos N Kutulakos, Hendrik Lensch, Marcus Magnor, and Wolfgang Heidrich. Transparent and specular object reconstruction. *CGF*, 29(8), 2010. 15

[9] Michael Oren and Shree K Nayar. A theory of specular surface geometry. *IJCV*, 24(2), 1997. 15, 21

[10] Stefan Roth and Michael J Black. Specular flow and the recovery of surface structure. In *CVPR*, 2006. 15

[11] Yair Adato, Yuriy Vasilyev, Ohad Ben-Shahar, and Todd Zickler. Toward a theory of shape from specular flow. In *ICCV*, 2007. 15

[12] Aswin C. Sankaranarayanan, Ashok Veeraraghavan, Oncel Tuzel, and Amit Agrawal. Image invariants for smooth reflective surfaces. In *ECCV*, 2010. 15

[13] Guillermo D Canas, Yuriy Vasilyev, Yair Adato, Todd Zickler, Steven Gortler, and Ohad Ben-Shahar. A linear formulation of shape from specular flow. In *ICCV*, 2009. 15

[14] Yair Adato, Todd Zickler, and Ohad Ben-Shahar. Toward robust estimation of specular flow. In *BMVC*, 2010. 15

[15] Yair Adato and Ohad Ben-Shahar. Specular flow and shape in one shot. In *BMVC*, 2011. 15

[16] Yuriy Vasilyev, Todd Zickler, Steven Gortler, and Ohad Ben-Shahar. Shape from specular flow: Is one flow enough? In *CVPR*, 2011. 15

[17] Thomas Bonfort and Peter Sturm. Voxel carving for specular surfaces. In *ICCV*, 2003. 16

[18] Silvio Savarese, Min Chen, and Pietro Perona. Recovering local shape of a mirror surface from reflection of a regular grid. In *ECCV*, 2004. 16

[19] Silvio Savarese, Min Chen, and Pietro Perona. Local shape from mirror reflections. *IJCV*, 64(1), 2005. 16

[20] Silvio Savarese, Li Fei-Fei, and Pietro Perona. What do reflections tell us about the shape of a mirror? *ACM TOG*, 2004. 16

[21] Marshall F Tappen. Recovering shape from a single image of a mirrored surface from curvature constraints. In *CVPR*, 2011. 16

[22] Jonathan Balzer, S Hofer, and Jürgen Beyerer. Multiview specular stereo reconstruction of large mirror surfaces. In *CVPR*, 2011. 16, 21

[23] Michael Weinmann, Aljosa Osep, Roland Ruiters, and Reinhard Klein. Multi-view normal field integration for 3D reconstruction of mirroring objects. In *ICCV*, 2013. 16, 21

[24] Tsuyoshi Nagato, Takashi Fuse, and Tetsuo Koezuka. Defect inspection technology for a gloss-coated surface using patterned illumination. In *IS&T/SPIE Electronic Imaging*, 2013. 16

[25] Miaomiao Liu, Richard Hartley, and Mathieu Salzmann. Mirror surface reconstruction from a single image. In *CVPR*, 2013. 16

[26] Marco Tarini, Hendrik P.A. Lensch, Michael Goesele, and Hans-Peter Seidel. *3D acquisition of mirroring objects*. MPI Informatik, Bibliothek & Dokumentation, 2003. 16

[27] Bastien Jacquet, Christian Häne, Kevin Köser, and Marc Pollefeys. Real-world normal map capture for nearly flat reflective surfaces. In *ICCV*, 2013. 16

[28] Diego Nehab, Tim Weyrich, and Szymon Rusinkiewicz. Dense 3D reconstruction from specularity consistency. In *CVPR*, 2008. 16

[29] Borom Tunwattanapong, Graham Fyffe, Paul Graham, Jay Busch, Xueming Yu, Abhijeet Ghosh, and Paul Debevec. Acquiring reflectance and shape

from continuous spherical harmonic illumination. *ACM TOG*, 32(4), 2013. 17

[30] Howard Schultz. Retrieving shape information from multiple images of a specular surface. In *PAMI*, volume 16, 1994. 17

[31] Zuoyong Zheng, Ma Lizhuang, Li Zhong, and Zhihua Chen. An extended photometric stereo algorithm for recovering specular object shape and its reflectance properties. *ComSIS*, 7(1), 2010. 17

[32] Hin-Shun Chung and Jiaya Jia. Efficient photometric stereo on glossy surfaces with wide specular lobes. In *CVPR*, 2008. 17

[33] Roland Ruiters and Reinhard Klein. Heightfield and spatially varying BRDF reconstruction for materials with interreflections. *CGF*, 28(2), 2009. 17

[34] Carlos Hernández, George Vogiatzis, and Roberto Cipolla. Multiview photometric stereo. *PAMI*, 2008. 17

[35] Zhenglong Zhou, Zhe Wu, and Ping Tan. Multi-view photometric stereo with spatially varying isotropic materials. In *CVPR*, 2013. 17

[36] Micah K Johnson and Edward H Adelson. Shape estimation in natural illumination. In *CVPR*, 2011. 17

[37] Geoffrey Oxholm and Ko Nishino. Shape and reflectance from natural illumination. In *ECCV*, pages 528–541, 2012. 17

[38] K. Nishino. Directional statistics BRDF model. In *ICCV*, 2009. 17

[39] G. Oxholm and K. Nishino. Multiview shape and reflectance from natural illumination. In *CVPR*, pages 2163–2170, June 2014. 17, 22, 26, 34, 36

[40] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 2002. 17, 68

[41] Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 2015. 17, 64

[42] René Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *CVPR*, 2016. 17

[43] Robert J Woodham. Photometric method for determining surface orientation from multiple images. *Optical engineering*, 1980. 18

[44] Austin Abrams, Christopher Hawley, and Robert Pless. Heliometric stereo: Shape from sun position. In *ECCV*, 2012. 18

[45] Jure Žbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 2016. 18, 58

[46] L'ubor Ladickỳ, Christian Häne, and Marc Pollefeys. Learning the matching function. *arXiv preprint arXiv:1502.00652*, 2015. 18

[47] W. Luo, A. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016. 18

[48] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 18, 46, 50

[49] Evan Shelhamer, Jonathon Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *PAMI*, 2016. 18, 46

[50] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 18

[51] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016. 18

[52] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *PAMI*, 2009. 18, 49, 56, 57, 60, 69, 74

[53] Derek Hoiem, Alexei A Efros, and Martial Hebert. Automatic photo pop-up. *TOG*, 2005. 19

[54] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *PAMI*, 2015. 19, 41, 52, 71

[55] L'ubor Ladickỳ, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *CVPR*, 2014. 19, 41

[56] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *PAMI*, 2014. 19, 55, 56, 57, 74, 84

[57] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 19, 41, 52, 53, 54, 60, 71

[58] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 19, 70

[59] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *CVPR*, 2015. 19

[60] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. *arXiv preprint arXiv:1605.02305*, 2016. 19

[61] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 2016. 19, 51, 55, 57, 74

[62] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *CVPR*, 2015. 19

[63] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016. 20

[64] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016. 20, 42, 43, 47, 48, 49, 53, 57, 60

[65] Ravi Garg, Vijay Kumar BG, and Ian Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 20, 43, 48, 52, 53, 54, 60, 64, 65, 68, 71, 73, 76

[66] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. *PAMI*, 16(2), 1994. 24

[67] Michael Kazhdan and Hugues Hoppe. Screened Poisson surface reconstruction. *ACM TOG*, 32(1), 2013. 24

[68] Simon Fuhrmann, Jens Ackermann, Thomas Kalbe, and Michael Goesele. Direct resampling for isotropic surface remeshing. In *VMV*, 2010. 24

[69] PJ Giblin, Frank E Pollick, and JE Rycroft. Recovery of an unknown axis of rotation from the profiles of a rotating surface. *JOSA A*, 1994. 25, 31

[70] Roberto Cipolla and Peter Giblin. *Visual motion of curves and surfaces*. Cambridge University Press, 2000. 25, 31

[71] George Vogiatzis, Carlos Hernandez, and Roberto Cipolla. Reconstruction in the round using photometric normals and silhouettes. In *CVPR*, 2006. 29

[72] Diego Nehab, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi. Efficiently combining positions and normals for precise 3D geometry. *ACM TOG*, 24(3), 2005. 31

[73] Christian Bloch. sIBL archive. www.hdrlabs.com/sibl/archive.html. 33

[74] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Adaptive structure from motion with a contrario model estimation. In *Asian Conference on Computer Vision*, pages 257–270. Springer, 2012. 33

[75] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5), 1976. 34

[76] Krzysztof M Gorski, Eric Hivon, AJ Banday, Benjamin D Wandelt, Frode K Hansen, Mstvos Reinecke, and Matthia Bartelmann. HEALPix: a framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622(2), 2005. 34

[77] Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. Optix: a general purpose ray tracing engine. *ACM TOG*, 29(4):66, 2010. 34

[78] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org. 34

[79] Kevin Karsch, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, Hailin Jin, Rafael Fonte, Michael Sittig, and David Forsyth. Automatic scene inference for 3d object compositing. *TOG*, 2014. 42

[80] Jonathan T Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. *CVPR*, 2015. 42

[81] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 2015. 42

[82] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose

recognition in parts from single depth images. *Communications of the ACM*, 2013. 42

[83] Danail Stoyanov, Marco Visentini Scarzanella, Philip Pratt, and Guang-Zhong Yang. Real-time stereo reconstruction in robotically assisted minimally invasive surgery. In *MICCAI*, 2010. 42

[84] Ian P Howard. *Perceiving in depth, volume 1: basic mechanisms*. Oxford University Press, 2012. 42

[85] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 44

[86] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 47, 49, 51, 52, 61, 69, 70, 71, 73, 77, 78

[87] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 47, 52, 53, 57, 72, 73, 76, 80

[88] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015. 47, 65

[89] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Is l2 a good loss function for neural networks for image processing? *arXiv preprint arXiv:1511.08861*, 2015. 48, 67

[90] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *Transactions on Image Processing*, 2004. 48, 67

[91] Philipp Heise, Sebastian Klose, Brian Jensen, and Alois Knoll. Pm-huber: Patchmatch with huber regularization for stereo matching. In *ICCV*, 2013. 48

[92] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 49, 57

[93] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 50

[94] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015. 50

[95] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 50

[96] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. http://distill.pub/2016/deconv-checkerboard/, 2016. 50

[97] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 50

[98] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 50, 70

[99] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 51, 69

[100] Miaomiao Liu, Mathieu Salzmann, and Xuming He. Discrete-continuous depth estimation from a single image. In *CVPR*, 2014. 55, 56, 57, 74

[101] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 2009. 57

[102] Derek Hoiem, Andrew N Stein, Alexei A Efros, and Martial Hebert. Recovering occlusion boundaries from a single image. In *ICCV*, 2007. 57

[103] Ahmad Humayun, Oisin Mac Aodha, and Gabriel J. Brostow. Learning to Find Occlusion Regions. In *CVPR*, 2011. 57

[104] Clément Godard, Peter Hedman, Wenbin Li, and Gabriel J Brostow. Multi-view reconstruction of highly specular surfaces in uncontrolled environments. In *3D Vision (3DV), 2015 International Conference on*, pages 19–27. IEEE, 2015. 58

[105] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 60, 61, 62, 64, 65, 67, 68, 69, 71, 72, 73, 74, 76

[106] Tinghui Zhou, Matthew Brown, Noah Snavely, and David Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 60, 62, 64, 65, 66, 70, 71, 72, 73, 74, 75, 76, 77, 78, 80

[107] Yevhen Kuznietsov, Jörg Stückler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017. 62, 71

[108] Yue Luo, Jimmy Ren, Mude Lin, Jiahao Pang, Wenxiu Sun, Hongsheng Li, and Liang Lin. Single view stereo matching. In *CVPR*, 2018. 62, 71

[109] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. *arXiv preprint arXiv:1709.06841*, 2017. 62, 70, 71

[110] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *CVPR*, 2018. 62, 70, 71, 73

[111] Arunkumar Byravan and Dieter Fox. Se3-nets: Learning rigid body motion using deep neural networks. In *ICRA*, 2017. 63

[112] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017. 63, 67, 70

[113] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018. 63, 65, 71, 73, 77, 78, 80

[114] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency. *arXiv preprint arXiv:1711.03665*, 2017. 63, 71

[115] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. *CVPR*, 2018. 63, 65, 66, 67, 71, 72, 73, 74, 77, 78, 80

[116] Chaoyang Wang, Jose Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. *CVPR*, 2018. 63, 65, 66, 68, 69, 71, 72, 73, 74, 75, 77, 78, 80

[117] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 65, 69

[118] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 69

[119] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015. 69

[120] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 70

[121] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 70

[122] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *AAAI*, 2017. 70

[123] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. *arXiv preprint arXiv:1709.02371*, 2017. 70

[124] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 70

[125] Jogendra Nath Kundu, Phani Krishna Uppala, Anuj Pahuja, and R. Venkatesh Babu. Adadepth: Unsupervised content congruent adaptation for depth estimation. *arXiv preprint arXiv:1803.01599*, 2018. 71

[126] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *Transactions on Robotics*, 2015. 78

[127] Fritz M Gavves E Tuytelaars T Rematas K, Ritschel T. Deep reflectance maps. *Proc. CVPR*, 2016. 84

[128] Danilo Jimenez Rezende, SM Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Advances In Neural Information Processing Systems*, pages 4996–5004, 2016. 84

[129] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object

reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 84

[130] Clément Godard, Kevin Matzen, and Matt Uyttendaele. Deep burst denoising. *arXiv preprint arXiv:1712.05790*, 2017. 84

[131] Daniel Zoran, Phillip Isola, Dilip Krishnan, and William T Freeman. Learning ordinal relationships for mid-level vision. In *ICCV*, 2015. 84

[132] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *NIPS*, 2016. 84

[133] Michael Firman, Oisin Mac Aodha, Simon Julier, and Gabriel J. Brostow. Structured Prediction of Unobserved Voxels From a Single Depth Image. In *CVPR*, 2016. 84

[134] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. *arXiv preprint arXiv:1804.00650*, 2018. 84