Combining Features and Semantics for Low-level Computer Vision

Combining Features and Semantics for Low-level Computer Vision

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Dipl.-Inform. Fatma Güney
aus Çanakkale, Turkei

Tübingen
2017

Tag der mündlichen Qualifikation:	16.11.2017
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Dr. Andreas Geiger
2. Berichterstatter:	Prof. Dr. Hendrik P. A. Lensch

Abstract

Visual perception of depth and motion plays a significant role in understanding and navigating the environment. Reconstructing outdoor scenes in 3D and estimating the motion from video cameras are of utmost importance for applications like autonomous driving. The corresponding problems in computer vision have witnessed tremendous progress over the last decades, yet some aspects still remain challenging today. Striking examples are reflecting and textureless surfaces or large motions which cannot be easily recovered using traditional local methods. Further challenges include occlusions, large distortions and difficult lighting conditions. In this thesis, we propose to overcome these challenges by modeling non-local interactions leveraging semantics and contextual information.

Firstly, for binocular stereo estimation, we propose to regularize over larger areas on the image using object-category specific disparity proposals which we sample using inverse graphics techniques based on a sparse disparity estimate and a semantic segmentation of the image. The disparity proposals encode the fact that objects of certain categories are not arbitrarily shaped but typically exhibit regular structures. We integrate them as non-local regularizer for the challenging object class 'car' into a superpixel-based graphical model and demonstrate its benefits especially in reflective regions.

Secondly, for 3D reconstruction, we leverage the fact that the larger the reconstructed area, the more likely objects of similar type and shape will occur in the scene. This is particularly true for outdoor scenes where buildings and vehicles often suffer from missing texture or reflections, but share similarity in 3D shape. We take advantage of this shape similarity by localizing objects using detectors and jointly reconstructing them while learning a volumetric model of their shape. This allows to reduce noise while completing missing surfaces as objects of similar shape benefit from all observations for the respective category. Evaluations with respect to LIDAR ground-truth on a novel challenging suburban dataset show the advantages of modeling structural dependencies between objects.

Finally, motivated by the success of deep learning techniques in matching problems, we present a method for learning context-aware features for solving optical flow using discrete optimization. Towards this goal, we present an efficient way of training a context network with a large receptive field size on top of a local network using dilated convolutions on patches. We perform feature matching by comparing each pixel in the reference image to every pixel in the target image, utilizing fast GPU matrix multiplication. The matching cost volume from the network's output forms the data term for discrete MAP inference in a pairwise Markov random field. Extensive evaluations reveal the importance of context for feature matching.

Kurzfassung

Die visuelle Wahrnehmung von Tiefe und Bewegung spielt eine wichtige Rolle bei dem Verständnis und der Navigation in unserer Umwelt. Die 3D Rekonstruktion von Szenen im Freien und die Schätzung der Bewegung von Videokameras sind von größter Bedeutung für Anwendungen, wie das autonome Fahren. Die Erforschung der entsprechenden Probleme des maschinellen Sehens hat in den letzten Jahrzehnten enorme Fortschritte gemacht, jedoch bleiben einige Aspekte heute noch ungelöst. Beispiele hierfür sind reflektierende und texturlose Oberflächen oder große Bewegungen, bei denen herkömmliche lokale Methoden häufig scheitern. Weitere Herausforderungen sind niedrige Bildraten, Verdeckungen, große Verzerrungen und schwierige Lichtverhältnisse. In dieser Arbeit schlagen wir vor nicht-lokale Interaktionen zu modellieren, die semantische und kontextbezogene Informationen nutzen, um diese Herausforderungen zu meistern.

Für die binokulare Stereo Schätzung schlagen wir zuallererst vor zusammenhängende Bereiche mit objektklassen-spezifischen Disparitäts Vorschlägen zu regularisieren, die wir mit inversen Grafik Techniken auf der Grundlage einer spärlichen Disparitätsschätzung und semantischen Segmentierung des Bildes erhalten. Die Disparitäts Vorschläge kodieren die Tatsache, dass die Gegenstände bestimmter Kategorien nicht willkürlich geformt sind, sondern typischerweise regelmäßige Strukturen aufweisen. Wir integrieren sie für die komplexe Objektklasse 'Auto' in Form eines nicht-lokalen Regularisierungsterm in ein Superpixel-basiertes grafisches Modell und zeigen die Vorteile vor allem in reflektierenden Bereichen.

Zweitens nutzen wir für die 3D-Rekonstruktion die Tatsache, dass mit der Größe der rekonstruierten Fläche auch die Wahrscheinlichkeit steigt, Objekte von ähnlicher Art und Form in der Szene zu enthalten. Dies gilt besonders für Szenen im Freien, in denen Gebäude und Fahrzeuge oft vorkommen, die unter fehlender Textur oder Reflexionen leiden aber ähnlichkeit in der Form aufweisen. Wir nutzen diese ähnlichkeiten zur Lokalisierung von Objekten mit Detektoren und zur gemeinsamen Rekonstruktion indem ein volumetrisches Modell ihrer Form erlernt wird. Dies ermöglicht auftretendes Rauschen zu reduzieren, während fehlende Flächen vervollständigt werden, da Objekte ähnlicher Form von allen Beobachtungen der jeweiligen Kategorie profitieren. Die Evaluierung auf einem neuen, herausfordernden vorstädtischen Datensatz in Anbetracht von LIDAR-Entfernungsdaten zeigt die Vorteile der Modellierung von strukturellen Abhängigkeiten zwischen Objekten.

Zuletzt, motiviert durch den Erfolg von Deep Learning Techniken bei der Mustererkennung, präsentieren wir eine Methode zum Erlernen von kontextbezogenen Merkmalen zur Lösung des optischen Flusses mittels diskreter Optimierung. Dazu stellen wir

eine effiziente Methode vor um zusätzlich zu einem Lokalen Netzwerk ein Kontext-Netzwerk zu erlernen, das mit Hilfe von erweiterter Faltung auf Patches ein großes rezeptives Feld besitzt. Für das Feature Matching vergleichen wir mit schnellen GPU-Matrixmultiplikation jedes Pixel im Referenzbild mit jedem Pixel im Zielbild. Das aus dem Netzwerk resultierende Matching Kostenvolumen bildet den Datenterm für eine diskrete MAP Inferenz in einem paarweisen Markov Random Field. Eine umfangreiche Evaluierung zeigt die Relevanz des Kontextes für das Feature Matching.

Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisor Dr. Andreas Geiger for his insightful guidance, remarkable patience, and continuous support. It has been a privilege to work with you.

I would like to thank the rest of my thesis committee: Prof. Hendrik Lensch, Prof. Michael Black, and Prof. Andreas Schilling for agreeing to be in my committee and for evaluating my work.

I am very grateful for all the opportunities and experience Max Planck Institute has provided me, but most of all, for all the great people in Perceiving Systems Department and Autonomous Vision Group. With a special mention to Thomas Nestmeyer, Joel Janai, Varun Jampani, Laura Sevilla, Christoph Lassner, Angjoo Kanazawa, Ali Osman Ulusoy, Martin Kiefel, Maren Mahsereci, Sergey Prokudin, Andreas Lehrmann, Silvia Zuffi, Federica Bogo, Naureen Mahmood, Cristina Garcia Cifuentes, Jonas Wulff, Peter Gehler, Raghudeep Gadde, Yiyi Liao, Aseem Behl, Lars Mescheder, Benjamin Coors, Despoina Paschalidou, Carolin Schmitt, Simon Donne, David Stutz, Gernot Riegler, and Moritz Menze. I am also grateful to the staff members who are very supportive and always ready to help: Jon Anning, Melanie Feldhofer, Magdalena Seebauer, Nicole Overbaugh, Telintor Ntouni, Sabrina Jung, Joan Piles, and Jojumon Kavalan. As Michael puts it very nicely: You all made me love coming to work every day!

Also, I would like to thank the Center for Learning Systems for half funding my PhD and for fun retreats at amazing locations.

I'd like to thank my friends in Turkey: Belma Engin, N. Çağrı Kılıboz, and D. Sinem Kılıboz, especially Beyza Ermiş for her amazing friendship and for sharing the whole experience.

And finally my mother, my father and my little sister: I'm eternally grateful for your unconditional support and for providing me the freedom to pursue the road less traveled. Thanks for all your encouragement!

Contents

Chapter 1

Introduction

The perception of depth and motion are two of the most basic aspects of visual perception. Since birth, we have a constant stream of visual stimuli that is interpreted by our brain as structure and motion. We naturally reason how far the surfaces and objects are located relatively and observe the changes in the visual input as we move in the world or when the other objects or agents move with respect to us as the observant. As soon as we are able to freely navigate in the world, the ability to perceive depth and motion become crucial. When we first enter a scene, we are very quick at grasping its characteristics of the close surrounding with all structures and entities and to identify possible obstacles and navigate safely. In other words, brain is capable of building complex 3D models of various scenes based on visual cues.

In this thesis, we introduce a similar notion for machine perception and propose algorithms for machines to perceive structure and motion. Despite the availability of sensors which can directly measure 3D information, visual processing using cameras remains important due to the high availability of cameras and their small cost. Visual perception of depth and motion is equally important for machines as for humans. There are various challenges in the related problems we tackle in computer vision. The perception in outdoor scenes is one particular challenge where autonomous driving is one of the most prominent applications. Estimating depth, creating reliable 3D models of the environment and estimating the motion of the vehicle itself as well as the other agents in the scene are of utmost importance for safe navigation. Given the importance of safety for these applications, machine perception needs to be perfected with a performance comparable to the human brain.

In computer vision, the perception of depth and motion tackles several problems including binocular stereo estimation from two cameras, multi-view reconstruction from several viewpoints, motion estimation in 2D and 3D. Over the last few decades, there has been tremendous progress in understanding and solving these problems. In the binocular case, we can obtain depth from correspondences by defining the geometry to relate two views simulating our two eyes. Given reliable depth estimates from more cameras observing a scene from multiple viewpoints or similarly, cameras moving in a scene, we can create 3D models of the scene in varying complexity or density. Yet, we haven't achieved fully established solutions for these problems; there are still many persistent,

Figure 1.1: **Matching Ambiguities.** This figure illustrates ambiguous cases for matching based problems. In the first case (**top**), the corresponding points encoded by color look very different on the left and right stereo image due to the reflective and transparent surfaces of cars. On the **bottom-left**, the problem with textureless surfaces is illustrated with a set of candidate matches on a fisheye image. As magnified next to the image, the green matching point has no distinctive information inside the patch which looks exactly the same with many non-matching red points. These kind of problems are frequently observed in both stereo and optical flow estimation (Geiger *et al.*, 2012; Butler *et al.*, 2012). Appearance change due to the occlusions (marked red, **bottom-right**) is another frequently observed problem. While the assumption of similar appearance holds most of the time (green patch), the matching points look completely different in case of occlusions, for example due to the motion of the hand (red patch).

remaining sources of error which hold their applications back from relying on visual perception alone.

In essence, both depth and motion estimation can be reduced to a correspondence search problem across different viewpoints or consecutive time frames. Correspondence search is typically based on the assumption of similar appearance, i.e. corresponding points are more similar than any other two points. This assumption holds most of the time and the small violations of it can be overcome using a regularization that models the smoothness or planarity of local structures in the world. However, there are certain cases that are either inherently ambiguous such as textureless or reflective regions or that become ambiguous when the appearance changes due to the camera or the objects in the scene or both being in motion leading to occlusions or motion blur (see Fig. 1.1). In either case, the assumption of similar appearance for corresponding pixels is violated.

Furthermore, these problems might occur consistently in a large region on the image rendering local regularizers ineffective. In this thesis, we investigate the most common sources of error for these problems and propose principled solutions which jointly exploit different types of cues.

In particular, we focus on one aspect which has hitherto been mostly ignored: the semantics of the scene. The semantics of the scene includes information about the structures and objects in the scene through their meaning to us humans. We define an object or a structure by associating a large amount of information to its name. This information could be related to the scene configuration with common layouts such as road, buildings and sky for a street scene or could be contextual by defining where an object is commonly observed in the scene such as cars on the road and buildings on two sides of the street. Furthermore, semantics includes information about the shape of objects which constitutes a very powerful prior through a representative model. These appearance cues which correspond to semantic regions and representative shape models can either be extracted from images using recognition algorithms or can be learned from examples that are frequently observed. We combine these appearance cues with geometric cues in a systematic way for improving depth estimation and 3D reconstruction. We point out the importance of contextual information for incorporating such cues and propose a learning-based approach which exploits context for estimating the motion of pixels.

At a very high level, our approaches are biologically inspired as explained next. More specifically, our motivation for incorporating larger context or semantics for these problems can be associated with a fundamental question that has long been discussed in visual cognition for human perception.

1.1 Biological Vision

From a biological point of view, the ability to perceive the moving world around us is perfected by evolution. According to several studies, visual processing related to the perception of depth and motion takes place in very early stages of the complete visual perception process. In other words, it is the fundamental step for many other visual tasks such as recognizing the scene and the objects in it and locating and tracking their movements. On the other hand, we develop a very good understanding of the semantics of the world. We collect years of evidence on how the world is structured, what the scenes are composed of and how things around us move. We build a general knowledge about the shapes and motion patterns of objects around us and we can directly resort to that associated knowledge by the name of the object without any visual cue. For example, we have a rough idea about the typical shape of a car with two or four doors where the upper half is mostly glass and we can adjust this general representation in our mind to match a specific car belonging to a brand. Furthermore, we know for sure that cars on the street move in a rigid way.

There is a large amount of prior knowledge that could potentially be used in the visual

perception of depth and motion. The question of whether this kind of information is utilized by the brain is relevant for computer vision to come up with methods which resemble human performance in these tasks. In the following, we focus on the perception of depth, where it occurs in the brain and what kind of cues are used. We also point out that there are similar discussions for the motion.

Biological perception of depth is called stereoscopic vision due to the fact that we receive the 3D world through two (horizontally separated) eyes. The images taken by each eye slightly differ and this difference is processed in the visual processing areas of the brain to create the perception of relative depth. More specifically, the two images captured by the left and right eye are matched and the horizontal shift between the corresponding points on two images is inversely proportional to depth. The perception of depth in the brain occurs in stages (Howard and Rogers, 2008). In the early stages, basic cues such as edges, direction of motion and color information are extracted in areas called V1 and V2. The information flow happens in successive iterations with forward and backward connections between different stages. Over two centuries, scientists have been asking questions about the stage or stages responsible for the correspondence problem to understand the process in the brain (Ramachandran and Rogers-Ramachandran, 2009). Furthermore, questions have been raised about the handling of false matches and how our brain avoids them to achieve correct matching.

The main question that we are interested in is whether stereo involves more than comparing pixels across the images of two eyes. In other words, is stereo interleaved with the other perception tasks such as image segmentation or shape perception or is there a clear ordering which suggests that depth perception precedes these other tasks? This question has first been addressed by Hermann von Helmholtz in the 19th century (von Helmholtz, 1867). He hypothesizes that the brain initially recognizes contours of objects and then performs a comparison of recognized forms for stereopsis. This is mainly motivated by the complexity of the correspondence problem prone to false matches. His hypothesis is later challenged by Bela Julesz by using random-dot patterns instead of photographs with line drawings (Julesz, 1971). Even without any contour or form to recognize, depth perception is achieved. He concludes that depth perception is realized as a point to point matching and precedes the detection of outlines or boundaries. His theory is further supported by the finding of Jack Pettigrew about binocular nerve cells that extract horizontal shifts for stereo in the earliest stage of binocular processing (Howard and Rogers, 2008). While the theory of pure bottom-up depth perception has been the predominant belief for decades inspiring traditional formulations of computer vision problems, it has been challenged in more recent studies.

Experiments with random dot stereograms prove that observers can perceive depth without the need for edge or surface information or even semantics, but this does not imply that edge detection or surface interpretation play no part in the perception of depth. In particular, several recent studies demonstrate the flexibility of the brain's visual perception areas. For example, using random dot stereograms, Ramachandran forms a square that is clearly visible through the texture (Ramachandran *et al.*, 1973). Instead of shifting

the dots from one eye to the other as Julesz does, he shifts the entire square. Despite uncorrelated random dots between two images, the square floats out when viewed through a stereoscope. Another way of showing the relationship between the depth and surface representation is through occlusion relations. Occlusions are considered as one of the most important monocular cues in depth perception. When an object occludes another, it is a clear sign of their relative depth ordering. Shimojo and Nakayama argue that occlusion relationships play an important role in binocular vision (Nakayama and Shimojo, 1990). By distorting half occlusions whose position and size contain information that can be used to infer depth, they observe the effect of losing patterns that are visible in natural viewing.

These experiments show that visual processing of stereo is more complex than just matching pixels across the images of two eyes. In some cases, recognition of shapes like the square in the experiment might precede measuring the shift across the eyes. Or occlusions as a result of object and surface configurations in the world influence the early stages of stereo matching. The brain uses all the available cues to achieve the perception of depth and some cues might be more useful than others in case of noise or distortion. There are studies which suggest that the process might be even more sophisticated, e.g., the brain extracts depth information and constructs 3D surfaces in a feedback loop. This is based on the experiments which require segmenting the image based on implied occlusions and hallucinating illusory contours to be used as cues in stereo matching (Ramachandran, 1986). We know that V1 contains cells detecting horizontal shifts for disparity, but the illusory contours from implied occlusions are extracted in V2 (Howard and Rogers, 2008). Although these findings suggest that there might be a feedback from V2 to V1, there is no physiological proof for this yet.

1.2 Computer Vision

As evidenced by the biological studies reviewed above, the perception of depth and motion is a complex visual inference task. There are possibly different types of visual cues involved in the process. In an attempt to replicate the visual process in the brain, in computer vision, depth and motion estimation have been traditionally considered as establishing correspondences across different viewpoints or time instances. The process is referred as low-level vision and typically limited to bottom-up processing which builds on matching cues called features.

Modularity of visual processing into different levels dates back to David Marr's influential theory from primal sketches to 3D models (Marr, 1983). In his framework, Marr identified three main representations of an image: a primal sketch which is concerned with local intensity changes, edges, lines, curves and boundaries; a 2.5D sketch for contour, orientation, depth and other properties of visible surfaces; and lastly a 3D model representation for three dimensional objects, allowing their recognition. According to Marr's theory, there is a clear separation between the early visual system consisting of

the first two representations and the last high-level representation. The early representation covers low-level factors affecting the intensities perceived, namely the geometry, the reflectance of the visible surfaces, the illumination and the viewpoint. In order to model the structure of the visible surfaces, Marr proposed to first capture local properties, so-called low-level features, which correspond to oriented edges, bars, ends and blobs. These viewpoint-dependent representations intend to describe the orientation and depth of the visible surfaces as well as discontinuities but do not describe the full shape information or the identity of the object which are covered by the last representation. This separation agrees with Julesz's demonstration on depth perception without any high-level information.

The literature in computer vision has mostly followed this representation by keeping low-level processing and high-level cues separate. The existing approaches for perceiving depth and modeling structures in 3D have mainly focused on describing pixels or local neighborhoods of pixels by assuming a local structure representing smooth surfaces in the world and finding efficient ways of searching the solution space. Despite the obvious problems of correspondence search based on local appearance alone and the huge progress made in other, related visual processing tasks such as object recognition and semantic segmentation, the role of semantics in low-level vision has remained mostly unexplored. In contrast to the studies in biological vision, in computer vision, the original theory of Helmholtz based on matching higher levels of forms, i.e. objects instead of pixels has attracted very little attention so far.

In this thesis, we do not propose to explicitly match objects instead of pixels, but rather we introduce and combine high-level information from semantics with existing approaches based on geometric cues. There is an abundance of prior knowledge which can be utilized to improve these problems in computer vision. We focus on very problematic cases that are frequently observed in street scenes such as reflective surfaces of cars and saturated, textureless surfaces of buildings. In our solutions, we detect or segment these objects to locate them and model their shape either by using external 3D CAD models as shape priors or by clustering similar shapes in the scene and benefiting from generalization of the resulting shape models. Inspired by Helmholtz's theory, we present an approach for resolving inherent ambiguities of the correspondence search problem in stereo vision, 3D reconstruction and optical flow estimation.

1.3 The Proposed Approach

Our approach can be summarized as improving matching based problems by extending reasoning beyond local interactions. This can be done either by incorporating semantics of the scene or by learning robust image features which leverage information from a large contextual region. In particular, we make the following contributions:

- We propose to combine low-level matching with high-level reasoning about 3D objects to improve binocular stereo matching (Güney and Geiger, 2015).
- We propose a method to exploit frequently observed object similarities in street scenes to increase the completeness of 3D reconstructions (Zhou *et al.*, 2015).
- We propose to improve feature matching by learning context-aware features for optical flow estimation (Güney and Geiger, 2016).

In the following, we briefly introduce each of these problems in computer vision and explain the proposed solution. We conclude this chapter with an outline of the thesis.

1.3.1 Stereo Estimation using Object Knowledge

Since David Marr's influential investigations into the visual processing (Marr, 1983), stereo matching has been considered primarily a low-level process that builds on bottom-up representations like Marr's primal sketch. The vast majority of binocular stereo matching algorithms that have been proposed in the literature rely on low-level features in combination with relatively simple first or second order smoothness assumptions about the world. Little is known about the importance of recognition for this problem and the leading entries in current benchmarks completely ignore semantic information. This is in stark contrast to single image depth estimation methods which can extract depth information based on monocular appearance cues. In this thesis, we formulate the following question: How important are Helmholtz's early observations von Helmholtz (1867) for binocular stereo matching after all?

Typically, this problem is posed as inference using a graphical model where nodes in the graph correspond to pixels or superpixels representing the 3D planes in the world (Yamaguchi *et al.*, 2012). The agreement in appearance between two images is specified by a data term with a distribution associated to each node and pairwise relationships between neighboring nodes encouraging smoothness in estimations. Enforcing agreement between neighboring regions helps to improve results where the data term is ambiguous by smoothing out noise. However, this kind of modeling is ineffective in large textureless or reflecting surfaces where there is very little information in the data term for the local smoothness prior to interpolate.

To tackle this problem, we investigate the utility of higher level processes such as object recognition and semantic segmentation for stereo matching. In particular, we focus

our attention on well-defined objects for which the data term is weak and current methods perform poorly, e.g. cars. Due to their textureless, reflective and semi-transparent nature, those object surfaces represent a major challenge for current state-of-the-art algorithms. In contrast, as humans we are able to effortlessly extract information about the geometry of cars even from a single image thanks to our intrinsic object knowledge and shape representation. Inspired by this fact, we introduce object knowledge for well-constrained object categories into a slanted-plane graphical model to improve estimations in these problematic regions.

We model long-range interactions using a set of plausible object disparity hypotheses which cover most likely configurations of an object conditioned on the image. First, we sample these hypotheses by leveraging inverse graphics given an initial semi-dense disparity estimate and a rough semantic segmentation of the image for the object category car. Then, we encourage the presence of these 2.5D shape samples in our formulation depending on how much their geometry and semantic mask agrees with the image observations. We perform extensive evaluations to demonstrate the effectiveness of the proposed framework for resolving problems in challenging cases with highly ambiguous information. At the same time, our method is able to extract 3D object representations which are consistent with the estimated disparity map and may serve as input to higher level reasoning.

1.3.2 3D Reconstruction by Exploiting Object Similarities

3D reconstruction is one of the fundamental problems in computer vision and has received a lot of attention over the last decades. Today's hardware capabilities allow for robust structure-from-motion pipelines, capable of reconstructing sparse 3D city models from millions of internet images (Frahm *et al.*, 2010) or dense models from video sequences in real time (Wendel *et al.*, 2012). With the advent of the Kinect sensor, depth information has become available indoors and approaches based on volumetric fusion (Nießner *et al.*, 2013) or mesh optimization (Zhou *et al.*, 2013) are able to produce reconstructions with fine details given a large number of RGB-D observations.

However, obtaining accurate reconstructions outdoors in less constrained environments, as observed during urban driving (Pollefeys, 2008; Gallup *et al.*, 2010b; Musialski *et al.*, 2013) remains highly challenging: RGB-D sensors (e.g., Kinect) cannot be employed outdoors due to their very limited range and expensive LIDAR-based mobile solutions result in relatively sparse 3D point clouds. Besides, reconstruction in uncontrolled environments can be very challenging due to difficult lighting conditions, occlusions and many objects which are only visible in a few frames.

We address these difficulties by taking advantage of an observation which has been largely ignored: The larger the reconstructed area, the more likely objects of similar type and shape (e.g., buildings or vehicles) will occur in the scene. Furthermore, man-made environments exhibit certain regularities and contain recurring structures like buildings with similar shapes. Yet, most of the existing work on reconstructing 3D scenes ignores

these structural dependencies between objects.

Inspired by the impressive capability of toddlers to learn about objects from very few observations (Biederman, 1987), we aim for minimal supervision and ask the following question: Can we exploit recurring objects of similar 3D shapes for jointly learning a shape model while reconstructing the scene? Such a system would be useful in many ways: First, we can expect increased completeness of the 3D reconstruction by regularizing across shapes using the principle of parsimony (i.e., by limiting the number of models and parameters). Second, we obtain a decomposition of a scene into its constituent objects in terms of 3D bounding boxes and segments. And finally, other tasks such as recognition or synthesis could benefit from the learned 3D shape models as well.

Our goal is to obtain a volumetric 3D reconstruction of the scene by leveraging the fact that 3D objects with similar shapes appear frequently in large environments. In particular, we start from a basic volumetric reconstruction pipeline which recovers a volumetric representation of the scene. We refer to this as initial reconstruction and use it as a baseline in our experiments. Based on this reconstruction, we apply an ensemble of exemplar-based object detectors to find instances of objects in the scene. Given the initial reconstruction and the object detections, we jointly cluster objects into categories according to their 3D shape, learn a 3D shape model for each of these categories and complete the 3D reconstruction by filling-in missing information from these jointly estimated 3D shape models. After the convergence of optimization, the information from learned shape models for the detected objects smoothly blends in with the initial reconstruction for the non-object parts of the scene such as road or sidewalk.

1.3.3 Learning Context-aware Features for Optical Flow

Despite large progress, optical flow is still an unsolved problem in computer vision. Challenges provided by autonomous driving applications or current benchmarks like KITTI and MPI Sintel include large motions, appearance changes, as well as uniform image regions. While the predominant paradigm for estimating optical flow is based on continuous optimization Horn and Schunck (1981); Lucas and Kanade (1981); Black and Anandan (1993) with coarse-to-fine warping Brox *et al.* (2004), recent approaches leverage discrete optimization strategies Menze *et al.* (2015); Chen and Koltun (2016); Wulff and Black (2015); Hornacek *et al.* (2014) in order to overcome local minima and to gain robustness.

While these approaches have shown promising results, their performance still falls considerably behind the state-of-the-art in stereo matching. One reason for this is that 2D flow estimation is an inherently more difficult problem than 1D matching along the epipolar line. Another reason might be that most existing works on discrete optical flow optimization still depend on hand-crafted features for calculating the matching costs. In contrast, the most successful approaches in stereo matching exploit a combination of learned local feature representations and global discrete optimization (Žbontar and LeCun, 2016; Chen *et al.*, 2015b; Luo *et al.*, 2016a).

There has been tremendous progress over the last years to describe the appearance of local image regions. Early approaches employed image descriptors based on hand-crafted aggregation strategies such as histogram of edge orientations Lowe (2004); Tola *et al.* (2010). Following the recent trend in computer vision (Krizhevsky *et al.*, 2012), there has been a shift from hand-crafted features to deep learning techniques in order to learn representations of image regions which are designed for a specific task. Specifically, Siamese networks which are composed of two identical branches to process and correlate two image regions have become very popular (Bromley *et al.*, 1993; Žbontar and LeCun, 2016).

In this thesis, we investigate the utility of feature learning for discrete optical flow. In particular, we modify the "DiscreteFlow" framework of Menze *et al.* (2015) by replacing their hand-crafted descriptors with learned feature representations. We investigate two types of networks: a local network with 3×3 convolutions resulting in a relatively small receptive field size as well as a subsequent context network that aggregates information over larger image regions using dilated convolutions (Yu and Koltun, 2016). As naïve *patch-based* training with dilated convolutions is computationally very expensive, we propose an efficient implementation based on regular strided convolutions. For efficient learning of the whole pipeline, we specify auxiliary loss functions and train the model in a piecewise fashion.

1.3.4 Organization

We first provide necessary foundations in Section 2. We describe the most relevant related work in Section 3. We present our solutions with extensive evaluations in Section 4, Section 5 and Section 6. Finally, we conclude with possible future work directions in Section 7.

In the foundations chapter, we introduce the problem of depth estimation in the context of both binocular and multi-view stereo. We define optical flow as a specific case of motion estimation in 2D. Subsequently, we give a brief summary of traditional approaches to these problems in computer vision literature. This goal of this chapter is to define the problems and review the most common approaches relating existing works to each other. In addition, we provide a brief summary of graphical models with a focus on specific inference techniques used in this thesis to introduce the necessary background. We leave the most relevant approaches to the following related work chapter.

The related work chapter mainly focuses on two lines of work: feature learning using deep learning and reconstruction using appearance cues or semantics in the context of both binocular and multi-view stereo. We include single image depth estimation methods as the extreme case and also refer to methods which follow a similar approach to ours from various domains such as tracking, segmentation, facial and medical image analysis.

Each of the three approaches proposed in this thesis is described in a separate chapter. We follow a common organization for each by first introducing the method and then presenting the experiments. For stereo estimation, we first explain how we sample a

representative set of 3D CAD models. Then, we formulate our solution by describing each part of the model in detail and finally we describe the inference procedure. For 3D reconstruction, we first introduce the data collected from omnidirectional cameras by focusing on particular challenges. We provide a brief description of the calibration process for the multi-rig setup including the projection model used. Then, we describe the initial volumetric reconstruction step as well as the voxel hashing technique used for efficiency. Finally, we present our joint formulation for learning shape models while reconstructing the scene and explain the details of the inference procedure. In the optical flow chapter, we first introduce Siamese networks for feature learning and dilated convolutions as an efficient way of increasing their receptive field size. Then, we introduce our approach for learning features for discrete optical flow estimation.

Chapter 2

Foundations

In this chapter, we introduce the foundations for stereo estimation, multi-view reconstruction and optical flow estimation. In particular, we follow an organization by first introducing traditional approaches as well as more recent established solutions such as deep learning. Our goal is to provide a big picture for each problem, especially by focusing on particular challenges that we tackle in this thesis such as outdoor scenes, local regularizers and large motions. In the last section (Section 2.4), we provide basics of inference techniques we used in our approaches including message passing algorithms and stochastic sampling methods.

In Section 2.1, we first define the classical stereo estimation from two cameras by explaining the epipolar geometry. Then, we introduce traditional approaches to stereo estimation by following the traditional classification of stereo methods as local and global. We review planarity-based approaches as a way of introducing our prior knowledge about the structures and end our discussion with more recent deep learning approaches. In the end, we introduce the two most commonly used benchmarks for stereo estimation.

In Section 2.2, we start the discussion on the 3D reconstruction with traditional structure from motion problem. We follow a representation-based classification for introducing general approaches in 3D reconstruction. Then, we limit our focus on the 3D reconstruction of urban scenes with specific challenges and examples of depth map fusion approaches. The planarity-based approaches as well as geometric primitives are introduced as a way of regularizing the reconstruction. We finally discuss omnidirectional vision with its specific challenges in the calibration as well as benefits in autonomous driving related applications with some examples.

In Section 2.3, we define the ill-posed problem of optical flow estimation by introducing two traditional approaches. We detail our discussion on various aspects of global approaches such as robustness, large displacements and different formulations. Then, we introduce sparse matching, discrete formulations and compact motion models including layered and learned subspace models. We finally discuss more recent approaches that make use of domain knowledge such as epipolar flow and semantic flow as well as deep learning approaches. In the end, we introduce commonly used datasets for optical flow including more recent synthetic datasets for training deep neural networks.

2.1 Stereo Estimation

Stereo estimation is the process of extracting 3D information from 2D images captured by stereo cameras. In particular, stereo algorithms estimate depth information by finding correspondences in two images taken at the same point in time, typically by two cameras mounted next to each other on a fixed rig. These correspondences are projections of the same physical surface in the 3D world.

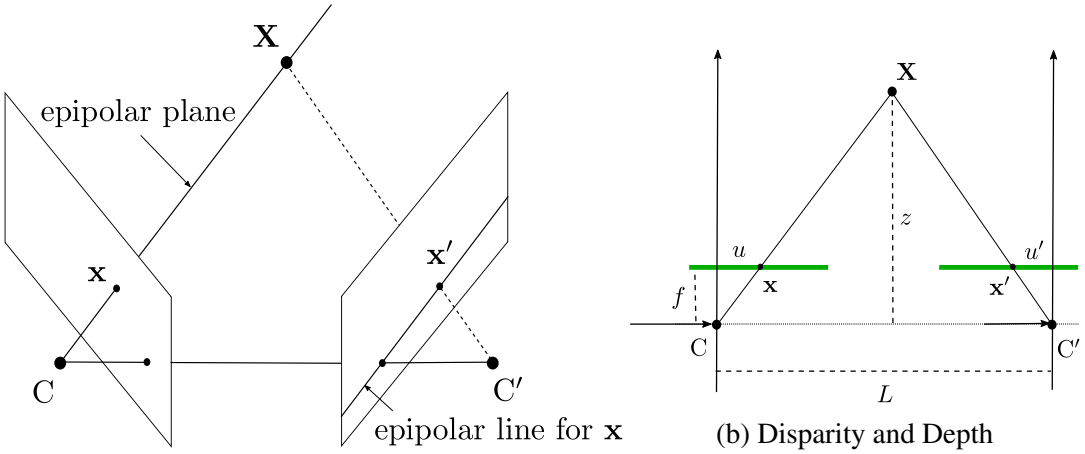
Stereo estimation finds applications in a wide range of areas including robotic navigation and manipulation, novel view synthesis, 3D model building and many more. Depth information is particularly crucial for applications in autonomous driving or driver assistance systems. The accurate estimation of dense depth maps is a necessary step for 3D reconstruction problems and many other problems such as obstacle detection, free space analysis and tracking benefit from the availability of reliable depth estimates.

2.1.1 Geometry of Stereo Matching

In the context of binocular stereo estimation, we limit our focus to calibrated two view geometry. In this setting, we assume that the camera geometry of the two views, i.e. their relative positions and their internal parameters, is known. We are interested in the correspondence problem: given an image point in the first view, what is the corresponding point in the second view? In this section, we first describe the underlying geometry and the rectification followed by the matching process, as commonly used by stereo camera setups.

Fig. 2.1(a) shows the stereo setup with two cameras and their image planes. Given a scene point $\mathbf{X}$ in 3D, the epipolar geometry relates its projection $\mathbf{x}$ in the first view to $\mathbf{x}'$ in the second (Hartley and Zisserman, 2004). The two camera centers and any point $\mathbf{X}$ in the scene define a plane called the epipolar plane. Rays back-projected from $\mathbf{x}$ and $\mathbf{x}'$ intersect at $\mathbf{X}$ and lie on the epipolar plane. The back-projection ray from an image point $\mathbf{x}$ in the first view projects to a line in the second view. This line is called the epipolar line corresponding to $\mathbf{x}$. Since $\mathbf{X}$ lies on the back-projection ray, $\mathbf{x}'$ must lie on the epipolar line. In terms of correspondence search, this constrains the position of the corresponding point $\mathbf{x}'$ to a line rather than the entire image.

Rectification is the process of applying 2D projective transformations to the images in order to match the epipolar lines. Images are transformed so that the epipolar lines are parallel to the horizontal axis and furthermore, matching points have the same vertical coordinate. The resulting rectified geometry leads to the concept of horizontal disparity d as inversely proportional to depth z. Assuming a calibrated, rectified stereo setup, disparity is the horizontal displacement of a point's projections between the left and the right image. As shown in Fig. 2.1(b), the relationship between disparity and depth can



(a) Epipolar Geometry

(b) Disparity and Depth

Figure 2.1: **Stereo Setup.** This figure shows the stereo setup with two cameras and their image planes. Given a scene point **X** in 3D, the epipolar geometry relates its projection **x** in the first view to $\mathbf{x}'$ in the second (Hartley and Zisserman, 2004). (a) shows the epipolar plane defined by the two camera centers and any point **X** in the scene. The back-projection ray from an image point **x** in the first view projects to the epipolar line corresponding to **x** in the second view. (b) shows a simplified version for deriving the relationship between disparity and depth after rectification.

be derived from similar triangles:

$$\frac{L}{z} = \frac{(L+u')-u}{z-f} \tag{2.1}$$

$$d = u - u' = f\frac{L}{z} \tag{2.2}$$

where f is the focal length, L is the baseline, and the disparity d relates the pixel $\mathbf{x} = (u,v)^T$ on the left image to $\mathbf{x}' = (u',v)^T$ on the right image.

After rectification, corresponding pixels **x** and $\mathbf{x}'$ can be compared over all possible disparity values d. Associated values which represent the confidence or likelihood of each disparity, i.e. cost, can be stored in a cost volume. In the stereo literature, the cost volume is often referred as the disparity space image (DSI) defined over a disparity space $(\mathbf{x},\mathbf{x}',d)$ (Szeliski, 2011). The goal of a stereo algorithm is to find the true disparity in that space. Next, we introduce traditional approaches for the stereo estimation problem by describing a set of representative methods proposed over the last decades.

2.1.2 Traditional Approaches

In computer vision, stereo estimation is one of the fundamental problems and it has been widely studied. In this section, we provide an overview of the existing approaches in order to show the evolution of stereo algorithms over the years. Multiple taxonomies for stereo matching have been proposed in the literature. Guided by the computational restrictions, the earliest one is based on the density of the output. Feature-based methods produce only sparse correspondences based on interest points or edge detectors. Seed-and-grow methods start from reliable features and grow into more matches, potentially followed by a surface fitting step in order to produce an estimate in all image regions. Although producing an accurate dense disparity map is challenging due to occlusions and textureless regions, it is a desired property in modern stereo algorithms, since dense estimates are required by many applications such as view synthesis and image-based rendering.

Here, we focus on dense approaches which output an estimate at every point. We first list the typical steps in a stereo algorithm as described in detail in Scharstein and Szeliski (2002). We give a summary of the most commonly used matching functions in the literature and point out their failure cases, as shown by the comparative study of Hirschmüller and Scharstein (2007). Following the literature on the classification of stereo algorithms based on the optimization technique, we refer to local and global methods with an emphasis on Semi-Global Matching (Hirschmüller, 2008) and its extensions. From pixel-level methods, we move to the segment-based methods which use piecewise planarity assumptions as a form of regularization and mention examples of both superpixel-based and variational approaches. Finally, we give an overview of recent deep learning approaches for stereo estimation.

As postulated by Scharstein and Szeliski (2002), a traditional stereo matching algorithm consists of four steps, namely the matching cost computation, the cost (support) aggregation, the disparity computation and optimization, and the disparity refinement. The traditional sum of sum-of-squared-differences (SSD) algorithm follows these steps. First, the matching cost is computed as the squared difference of intensity values at a given disparity. Then, in the aggregation step, matching costs are summed over a window with constant disparity, and disparities are computed by selecting the minimal aggregated value at each pixel which is known as the winner-takes-all (WTA) solution. Algorithms often combine the first two steps and compute a matching cost based on a support region, such as normalized cross-correlation (NCC).

Matching Cost Function

Stereo matching is a correspondence problem, i.e. the goal is to identify matching points between the left and the right image, typically based on a cost function. The matching cost computation is the process of computing a cost function at each pixel for all possible disparities which ideally takes its minimal value at the true disparity. However, it is

hard to design such a cost function in practice. The simplest assumption used by stereo algorithms is the brightness constancy for corresponding pixels, which leads to absolute or squared differences. There are more robust cost functions which use truncated versions of these differences or use a support region to compute the cost instead of a single pixel, i.e. window-based cost functions. Some of the window-based cost functions can be computed using filters such as the rank filter, the Laplacian of Gaussian (LoG) or the mean filter. There are more complicated cost functions proposed in the literature such as the mutual information, the census transform and the entropy.

The assumption of brightness constancy is often violated in real-world situations, such as cameras with slightly different settings causing exposure changes, vignetting, image noise, non-Lambertian surfaces, illumination changes, etc. Hirschmüller and Scharstein (2007) call these changes radiometric differences and systematically investigate their effect on commonly used matching cost functions: sum of absolute differences, filter-based costs (LoG, Rank and Mean), hierarchical mutual information (HMI), and normalized cross-correlation (NCC). Although the performance of a cost function depends on the stereo method that uses it, these changes consistently degrade the performance of methods. On images with simulated and real radiometric differences, the rank filter performed best for correlation-based methods. For global methods, in tests with global radiometric changes or noise, HMI performed best, while in the presence of local radiometric variations, rank and LoG filters performed better than HMI. Qualitative results show that filter-based cost functions cause blurred object boundaries when used with global methods. None of the matching costs evaluated could succeed at handling strong lighting changes.

Local and Global Methods

A commonly referred taxonomy of stereo algorithms is based on the optimization being either local or global. Local methods focus on matching cost computation and aggregation by summing or averaging over a support region in the DSI. The final disparities are computed by simply selecting the lowest matching cost, i.e. the WTA solution. Matching ambiguities is a common source of error for local methods. Furthermore, local methods cannot model long-range interactions and fail to handle cases which require complex reasoning such as occlusions.

Global methods formulate disparity computation as an energy minimization problem based on an energy function typically composed of a data term and a smoothness term between neighboring pixels or regions. The corresponding graphical model of a global function with common smoothness functions is loopy by nature, therefore finding the global minimum in such a model is an NP-hard optimization problem. There is a variety of algorithms to compute approximate solutions such as graph-cuts, loopy belief propagation or mean field approximations. Another global optimization technique is dynamic programming that can find the global minimum for independent scanlines in polynomial time. However, the optimization in one dimension for each scanline individually leads

to streaking effects. A better approach is to evaluate the cumulative cost function from multiple directions as explained next.

Semi-Global Matching

Semi-Global Matching (SGM) (Hirschmüller, 2008) is a global optimization algorithm based on dynamic programming. The energy function has two levels of penalization for small and large disparity differences. The energy is calculated by summing costs along 1D paths from multiple directions towards each pixel using dynamic programming. The resulting disparity estimate is determined by the WTA. SGM has become very influential due to its speed and high accuracy as evidenced in various benchmarks such as Middlebury (Scharstein and Szeliski, 2002) or KITTI (Geiger *et al.*, 2012). SGM has been recently used on top of CNN features (Žbontar and LeCun, 2016; Luo *et al.*, 2016a).

There are a couple of follow-up works investigating the practical and theoretical sides of SGM. Gehrig *et al.* (2009) propose a real-time, low-power implementation of the SGM with algorithmic extensions for automotive applications on a reconfigurable hardware platform. Drory *et al.* (2014) offer a principled explanation for the success of SGM by clarifying its relation to belief propagation and tree-reweighted message passing with an uncertainty measure as outcome. The performance of SGM can be further improved by incorporating confidences into the stereo estimation (Seki and Pollefeys, 2016) or by using a CNN to learn penalties (Seki and Pollefeys, 2017).

Planarity

The inherent ambiguity in appearance based matching costs can be overcome by regularization, i.e., by introducing prior knowledge about the expected disparity map into the stereo estimation process. The simplest prior favors neighboring pixels to take the same disparity value, i.e. first order priors. The order of a smoothness prior is defined by the order of the derivative which the prior regularizes. However, such generic smoothness priors fail to reconstruct poorly textured and slanted surfaces, as they favor fronto-parallel planes.

A more generic approach to handle arbitrary smoothness priors is using higher-order connections such as second order priors. Higher order priors are able to express more realistic assumptions about depth images, but are usually harder to optimize and computationally more demanding. Second order smoothness terms penalize non-zero second derivatives, encouraging planar depth maps (Woodford *et al.*, 2009). In order to optimize the resulting non-submodular graph with triple cliques, Woodford *et al.* (2009) repeatedly merge proposal depth maps using an extension of α-expansion.

Piecewise planarity can be explicitly modeled by enforcing the image regions to be planar. Segment-based methods assume that the scene structure can be approximated by a set of non-overlapping planes. Given an over-segmentation of the reference image, disparities are estimated for each region which corresponds to a plane. Typically, algorithms

generate a set of planar hypotheses for each segment by applying plane fitting to an initial disparity map, and optimize over the hypotheses in a discrete labeling framework using graph-cuts or loopy BP (Klaus *et al.*, 2006). Refining the shape of the segments as part of the stereo optimization process leads to improved results as shown by Taguchi *et al.* (2008). In order to recover from large segmentation errors, Bleyer *et al.* (2010) model segmentation as a soft constraint where each pixel is assigned to a 3D surface (planes or B-splines). In contrast to the disparity representation of Woodford *et al.* (2009), the surface representation as a continuous 3D surface model allows for an easy incorporation of minimum description length (MDL) and soft segmentation priors.

Segment-based approaches usually have much smaller number of parameters than pixel-level approaches. While this prevents over-fitting and reduces computational cost, segment-based approaches struggle with boundary adherence, i.e. a segment containing two surfaces which are not coplanar and with curved surfaces. Existing approaches typically use atomic units such as superpixels to address the first problem. Given a fixed set of superpixels on the reference image, Yamaguchi *et al.* (2012) model the surface under each superpixel as a slanted plane and jointly reason about occlusion boundaries and depth in a hybrid MRF composed of both continuous and discrete random variables. Zhang *et al.* (2014a) combine second order and segment-based priors in a unified framework where a segment is modeled by a quadratic surface to allow for curved surfaces.

Planarity in Local Approaches: Local algorithms have also adapted the idea of piecewise planarity. Window-based local algorithms typically assume that all pixels within the support window have constant disparity. This assumption is violated when the support window contains pixels that lie on a different surface than the center pixel. While adaptive window sizes could be a solution for that (Hosni *et al.*, 2009; Rhemann *et al.*, 2011; Richardt *et al.*, 2010; Yoon and Kweon, 2005), the assumption is still wrong in case of slanted surfaces. Rather than applying fronto-parallel windows at discrete disparities, Michael Bleyer and Rother (2011) estimate a slanted support window at each pixel. Their algorithm is based on the PatchMatch idea of random search and propagation (Barnes *et al.*, 2009) to find the nearest neighbor on the epipolar line according to a plane. Geiger *et al.* (2010) use piecewise planarity to reduce the search space and gain efficiency. They build a prior over the disparity space by forming a triangulation on a set of robustly matched correspondences, called support points. A piecewise linear function is used to interpolate the disparities using the triangulation computed on the support points. This reduces matching ambiguities and results in an efficient algorithm by restricting the search to plausible regions.

Variational Approaches: In variational approaches, a commonly used smoothness prior, the Total Variation (TV) does not produce convincing results in the presence of weak and ambiguous observations, since it encourages piecewise constant regions leading to stair-casing artifacts. Häne *et al.* (2012) introduce patch-based priors into a TV

framework in the form of small, piecewise planar dictionaries. Total Generalized Variation (TGV) (Bredies *et al.*, 2010) is argued to be a better prior than TV, since it does not penalize piecewise affine solutions. However, it is restricted to convex data terms in contrast to TV, where global solutions can be computed even in the presence of non-convex data terms. Coarse-to-fine approaches as an approximation to the non-convex problem of stereo matching often end up with loss of details. To preserve fine details, Kuschk and Cremers (2013) integrate an adaptive regularization weight into the TGV framework by using edge detection and report improved results compared to a coarse-to-fine approach. Ranftl *et al.* (2013) obtain even better results by proposing a decomposition of the non-convex functional into two subproblems which can be solved globally where one is convex, and the other can be made convex by lifting the functional to a higher dimensional space.

Deep Learning

In the last years, deep learning approaches have gained popularity in stereo estimation. We leave the discussion for the ones about learning similarity functions (Žbontar and LeCun, 2016; Chen *et al.*, 2015b; Luo *et al.*, 2016a) and single image depth estimation methods for later (Section 3.1) and focus on other deep learning learning approaches here. Mayer *et al.* (2016) propose DispNet, an end-to-end approach by adapting the encoder-decoder architecture proposed by Dosovitskiy *et al.* (2015) that was used for optical flow estimation (see Section 2.3.1). The encoder computes abstract features while the decoder reestablishes the original resolution with additional cross-links between the contracting and expanding network parts. This kind of approach does not explicitly benefit from restrictive geometry of the stereo estimation and shows worse performance on the benchmarks compared to matching based approaches.

In contrast to using CNNs only for learning correspondences, there are some approaches which learn confidences or mimic energy minimization frameworks using a CNN. Seki and Pollefeys (2016) leverage CNNs to predict the confidences which are incorporated into the SGM formulation by weighting each pixel according to the estimated confidence value. Instead of a piece-wise approach, Seki and Pollefeys (2017) propose to directly learn penalties for SGM with loss functions resembling the SGM objective function. During training, the network is trained to minimize a path cost as well as a neighbor cost. In addition, they introduce a new parametrization that separates the positive and negative disparity changes. Experiments with the learned penalties and the new parametrization show improved results by learning discriminative representations of object structures. While Seki and Pollefeys (2016, 2017) start from a cost volume, e.g. created by Žbontar and LeCun (2016), Kendall *et al.* (2017) first form a cost volume using deep feature representations and then learn to incorporate contextual information using 3D convolutions over this volume. The model is trained end-to-end to regress disparities directly from the cost volume using a proposed differentiable soft argmin operation.

2.1.3 Datasets

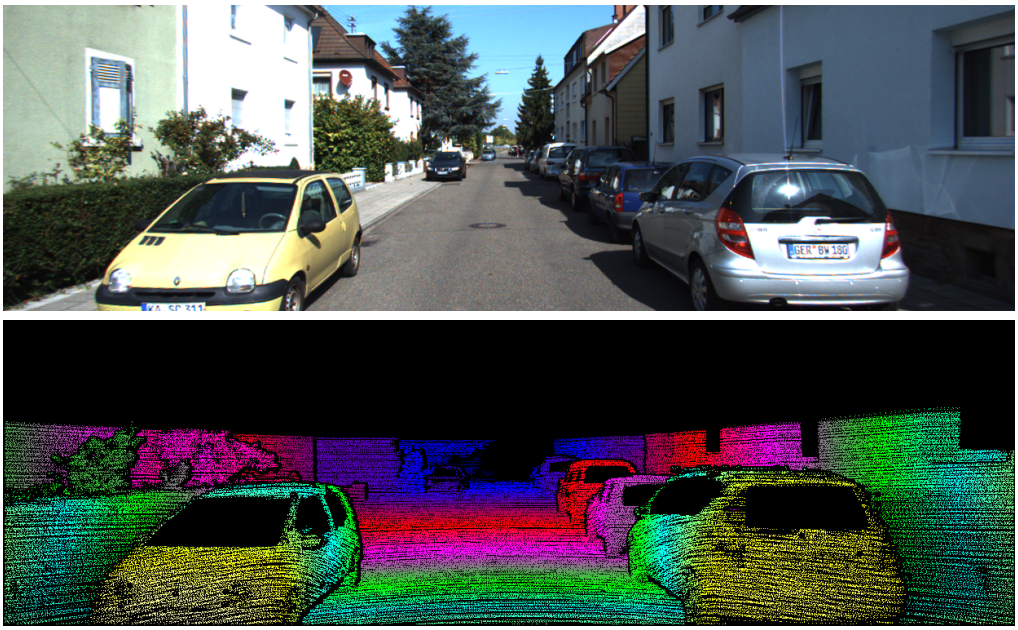


Figure 2.2: **Stereo Visualization.** This figure shows an image from the KITTI dataset (Geiger *et al.*, 2012) and the false coloring of the corresponding sparse ground-truth disparity map. The color from white to blue encodes the distance of the surfaces from the cameras.

In this section, we introduce two popular benchmarks for stereo estimation, the Middlebury and the KITTI, together with the common evaluation criteria used as well as ways of visualizing and comparing results. The Middlebury Benchmark uses Root Mean Squared (RMS) error measured in disparity units and the percentage of bad matching pixels based on a threshold, i.e. the outlier ratio. On the KITTI dataset, only the outlier ratio is used. Both benchmarks provide rankings based on different thresholds in occluded and non-occluded areas separately. In the qualitative evaluations, we use the false coloring used by the KITTI dataset (see Fig. 2.2).

The first dataset for evaluating stereo algorithms in a systematic way is the Middleburry Benchmark[1] (Scharstein and Szeliski, 2002, 2003; Scharstein and Pal, 2007; Hirschmüller and Scharstein, 2007; Scharstein *et al.*, 2014). The original dataset with 9 images (Scharstein and Szeliski, 2002) contains only piecewise planar scenes due to the piecewise planar technique used for obtaining ground-truth disparities. Scharstein and Szeliski (2003) use structured lighting technique to acquire high-complexity stereo image pairs with pixel-accurate ground-truth disparities. Using the same approach, the dataset is further extended to 30 stereo pairs for learning the parameters of probabilistic

[1] `http://vision.middlebury.edu/stereo/`

models (Scharstein and Pal, 2007) and 6 more with different exposures and different configurations of the light sources for evaluating the effect of these radiometric differences on stereo algorithms (Hirschmüller and Scharstein, 2007). Scharstein *et al.* (2014) extend the structured light technique by also modeling projector lens distortion and improving the rectification accuracy using the initial correspondences. This leads to sub-pixel accuracy with 33 new high-resolution pairs. These datasets are simplistic both in terms of the number of images and the scenes taken in a controlled environment, lacking realistic outdoor conditions.

The KITTI Vision Benchmark[2] is a standard testbed for stereo as well as for optical flow and other autonomous driving related tasks (Geiger *et al.*, 2012, 2013). The dataset has been captured from an autonomous driving platform and comprises six hours of recordings using high-resolution color and grayscale stereo cameras, a Velodyne 3D laser scanner and a high-precision GPS/IMU inertial navigation system. The stereo and optical flow benchmarks derived from this dataset comprise 194 training and 195 test image pairs at a resolution of 1280×376 pixels and sparse ground-truth obtained by projecting accumulated 3D laser point clouds onto the image. Due to the limitations of the rotating laser scanner used as reference sensor, the stereo and optical flow benchmark is restricted to static scenes with camera motion. Menze and Geiger (2015) have annotated 400 dynamic scenes, fitting accurate 3D CAD models to moving vehicles in to order to obtain flow and stereo ground truth for dynamic objects.

[2] `http://www.cvlibs.net/datasets/kitti/`

2.2 3D Reconstruction

Multi-view 3D reconstruction is the process of inverting the image formation process in order to model the underlying 3D geometry. In contrast to two-view stereo, multi-view reconstruction algorithms in particular address the problems of varying viewpoints and the complete reconstruction of 3D scenes from more than two and potentially a very large number of images.

In multi-view reconstruction, the goal is to produce a geometric representation of the underlying 3D world from a given set of images. The underlying principle is that a 3D point looks similar when projected to different viewpoints. Therefore, the classical approach is to find dense correspondences in images and triangulate to obtain a 3D reconstruction. The camera poses and the projection models of the cameras need to be known or else estimated for defining projections from 3D world points to 2D images. There exist various geometric representations such as disparity maps, a 3D point cloud or volume with voxel occupancies and signed distance fields.

A particular challenge is the reconstruction of urban scenes from moving stereo cameras as in the context of autonomous driving. A common approach for this problem is to estimate dense depth maps from nearby viewpoints observing the same scene and then fuse them into a common representation such as a triangular mesh or an implicit volumetric representation. In order to deal with the problems of matching or high computational cost, there are various approaches using local planarity assumptions or simple, global priors on the orientations of surfaces. A particular approach is to decompose scene into predefined 3D geometric primitives using simple shapes as building blocks. Omnidirectional cameras with a large field of view provide promising directions for solving these challenges in 3D reconstruction despite the additional challenges in their calibration.

In this section, we first introduce the Structure from Motion problem for calibrating cameras with examples of typical challenges for SfM approaches (Section 2.2.1). We also briefly explain general concepts such as the fundamental matrix, bundle adjustment and the visual odometry. In Section 2.2.2, we provide a general overview 3D reconstruction approaches based on the scene representation. Since we are mainly interested in the recontruction of urban scenes from omnidirectional cameras, we focus on traditional ways of solving these problems in the remainder of the section. In particular, we introduce challenges of urban reconstruction with a focus on depth map fusion, planarity assumptions and the 3D primitives in Section 2.2.2. Finally, we introduce the omnidirectional cameras, their calibration and a few applications in Section 2.2.2.

2.2.1 Structure from Motion

In this section, we provide a broad overview of the fundamentals of stereo vision. For a detailed description of concepts and algorithms mentioned, please refer to Hartley and Zisserman (2004) and Szeliski (2011). In contrast to Section 2.1, the descriptions are generalized to the multi-view case without assuming a known camera geometry. The

goal of structure from motion (SfM) is to simultaneously recover the camera geometry as well as the 3D structure from image correspondences. The camera geometry includes intrinsic properties such as the focal length under an assumed camera projection model as well as the camera pose which defines the location and orientation of the camera in the world. Multi-view stereo algorithms are highly dependent on the accurate estimation of the camera geometry due to the assumed epipolar geometry which simplifies the 2D matching problem to a 1D matching problem.

Each camera defines a projection from the 3D points in the world to the 2D points on its image plane:

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R} \mid \mathbf{t}] \tag{2.3}$$

Here, $\mathbf{K}$ denotes the intrinsic calibration matrix and $\mathbf{R}$ and $\mathbf{t}$ denote the extrinsic calibration parameters that define the position of the camera with respect to a reference frame. $\mathbf{R}$ is the 3×3 rotation matrix and $\mathbf{t} = -\mathbf{R}\,\mathbf{c}$ is the translation vector where $\mathbf{c}$ is the camera center. As a result, $\mathbf{P}$ is a 3×4 projection matrix which maps 3D points in the world to 2D points on the image plane

$$\mathbf{x} = \mathbf{P}\,\mathbf{X} \tag{2.4}$$

where $\mathbf{X} = (x, y, z, 1)^T$ is a 3D point in homogeneous coordinates and $\mathbf{x} = (u, v, 1)^T$ is the corresponding 2D point on the image.

Intrinsic and extrinsic parameters of the cameras are typically unknown and need to be estimated. This is known as the camera calibration step in a reconstruction pipeline and typically performed by using checkerboard patterns as fiducial markings. The parameters are obtained by minimizing the reprojection error between known pattern and the obtained projection. When there is more than one camera, i.e. the multi-camera setup, extrinsic calibration becomes more important to relate the points between different cameras and to represent them in a common coordinate system.

As explained in Section 2.1.1, epipolar geometry relates two views to each other. In the general case, this relation is defined by the fundamental matrix $\mathbf{F}$ which is a 3×3, rank 2 matrix:

$$\mathbf{x}_1^T\,\mathbf{F}\,\mathbf{x}_2 = 0 \tag{2.5}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ are two corresponding points in the two images. For two-frame reconstruction, the eight-point algorithm using point normalization is a well-known technique for computing the fundamental matrix (Hartley and Zisserman, 2004). In case of known intrinsic camera parameters, estimating the fundamental matrix is reduced to estimating the essential matrix $\mathbf{E}$ which can be solved using the five-point algorithm in conjuntion with RANSAC (Nistér, 2004).

SfM is the process of estimating the intrinsic and extrinsic parameters as well as the positions of the 3D points. In a typical framework, the first step is to detect sparse feature points for each image and match them to establish correspondences across multiple images. In video sequences, these are referred as feature tracks where each feature point in a track depends on the camera pose. SfM is typically carried out in an incremental

way by starting from an initial image pair and adding consecutive frames one by one. Images are sequentially processed for extracting and triangulating new 3D points while updating the whole 3D point cloud. For further accuracy, the SfM model is refined in a bundle adjustment step using all images. Bundle adjustment is a non-linear least-squares optimization process to compute camera parameters together with the positions of 3D points by minimizing the reprojection error.

Challenges: There are various aspects to the SfM problem in different contexts, each focusing on different challenges such as large scale or real-time processing. For example, SfM can be applied to large, unordered sets of images for reconstructing cities from millions of images (Frahm *et al.*, 2010) or even worldwide as in the Google Street View project for correcting camera poses from moving rolling shutter cameras[3] (Klingner *et al.*, 2013). While SfM is general enough to handle unordered sets of images, there is a related line of work called visual SLAM which specializes in processing ordered sequence of images in real-time. There are examples of real-time reconstruction pipelines for indoor scenes using a single camera (Klein and Murray, 2007; Newcombe and Andrew, 2010; Newcombe *et al.*, 2011a) or Kinect sensors for fusing multiple depth maps without RGB (Newcombe *et al.*, 2011b).

Outdoor scenes such as large-scale environments in autonomous driving are typically more complex and more challenging for real-time processing. Kinect sensors cannot be employed due to their limited range. One particular challenge is ensuring global consistency by identifying loop closures, i.e. relocalization.

Visual Odometry: Visual odometry (VO) is a specialized case of SfM aiming for local consistency without detecting loop closures. The computation time of SfM grows with the number of images due to the bundle adjustment step which refines the final structure and camera poses all together. VO is based on incrementally estimating the pose of the vehicle by typically processing only over a certain number of poses, i.e. windowed bundle adjustment. In contrast to visual SLAM, loop closure detection is not part of VO, but typically handled separately in a pose graph optimization step. VO can be used in combination with other measurements such as the wheel odometry and GPS.

2.2.2 Traditional Approaches

After the cameras are calibrated, solving for the 3D geometry of the scene is equivalent to solving the correspondence problem, based on a photo-consistency function which measures the agreement between different viewpoints. In the following steps, modern

[3]Rolling shutter cameras are a common type of generalized camera based on CMOS sensors with a rolling shutter that exposes the image sequentially in scanlines. A particular challenge for moving rolling shutter cameras is the undesired distortion effect, known as the rolling shutter effect.

reconstruction algorithms use dense matching rather than sparse matches in order to obtain detailed surfaces in the scene. Several categorizations of multi-view reconstruction algorithms have been proposed in the literature, typically considering the form of the photo-consistency function, the scene representation, visibility computation, priors, and initialization requirements as explained in detail in the survey for 3D reconstruction by Seitz *et al.* (2006).

Representations

From an application perspective, the scene representation is a common way of classifying multi-view reconstruction approaches into depth map, point cloud, mesh and volumetric. In the following, we briefly explain each representation with references to the representative work in the literature.

Depth Map: The depth map representation typically consists of a depth map for each input view estimated with a 3D modeling pipeline which starts with image matching followed by the pose estimation and dense stereo. This representation is usually preferred in scene analysis due to its flexibility and scalability to large scenes. One strategy which is particularly effective for urban scenes is Plane Sweeping Stereo algorithm (Collins, 1996). It sweeps a family of parallel planes in a scene, projects images onto a plane via planar homographies, then evaluates photo-consistency values on each plane. In large scenes, one challenge is to handle massive amount of data in real-time. Pollefeys (2008) propose a complete 3D reconstruction pipeline based on depth map representation. The real-time performance is achieved by incorporating a set of components which are particularly efficient on typical urban scenes such as a 2D feature tracker with automatic gain adaptation for handling large dynamic range in natural scenes and parallel implementations of plane sweeping stereo and depth map fusion on GPU.

Point-cloud: In contrast to partial depth maps which need to be merged into a global model, point-cloud or patch-based surface representations reconstruct a single 3D point-cloud model using all the input images. Under spatial consistency assumptions, the point-cloud on the surface of the scene can grow or expand which provides easy model manipulation such as merging and splitting. The representative work for these kind of approaches is Patch-based Multi-View Stereo (PMVS) by Furukawa and Ponce (2010). PMVS starts with a feature matching step to generate a sparse set of patches and then iterate between a greedy expansion step and a filtering step to make patches dense and remove erroneous matches. Another example is Goesele *et al.* (2007) which iteratively grows surfaces from 3D points reconstructed from SfM. These methods usually suffer from a large amount of parameters that need to be tuned for optimal results.

Volumetric: Volumetric approaches represent geometry on a regularly sampled 3D grid, i.e. volume, either as a discrete occupancy function (Kutulakos and Seitz, 2000) or a function encoding distance to the closest surface (level-set) (Faugeras and Keriven, 1998). More recent approaches use a probability map defined at regular voxel locations to encode the probability of occupancy (Bhotika *et al.*, 2002; Pollard and Mundy, 2007; Ulusoy *et al.*, 2015). The amount of memory required is the main limitation for volumetric approaches. There is a variety of methods for dealing with this problem such as voxel hashing (Nießner *et al.*, 2013) or a data adaptive discretization of the space in the form of a Delaunay triangulation (Labatut *et al.*, 2007). One effective solution is an octree data structure which is essentially an adaptive voxel grid to allocate high resolution cells only near the surfaces (Steinbrucker *et al.*, 2013). Steinbruecker *et al.* (2014) further accelerate processing by proposing an octree data structure that supports volumetric multi-resolution 3D mapping as well as an efficient incremental meshing algorithm.

Mesh or Surface: The final representation in reconstruction is typically triangular mesh-based surfaces. Volumetric surface extraction fuses 3D information from an intermediate representation such as depth maps, point clouds, volumes or scans into a single, clean mesh model. Seminal work by Curless and Levoy (1996) proposes an algorithm to accumulate surface evidence into a voxel grid using signed distance functions. The surface is implicitly represented as the zero crossing of the aggregated signed distance functions. It can be extracted using the Marching Cube algorithm (Lorensen and Cline, 1987) or using volumetric graph-cuts to label each voxel as interior or exterior. There are approaches which directly start from images and refine a mesh model using an energy function composed of a data term based on a photo-consistency function and a regularization term for smoothness. In these approaches, the energy is usually optimized using gradient descent, where the movement of each vertex is determined by the gradient of the objective function.

Urban Reconstruction

The goal of urban reconstruction algorithms is to produce fully automatic, high-quality, dense reconstructions of urban areas by addressing inherent challenges such as lighting conditions, occlusions, appearance changes, high-resolution inputs and large-scale outputs. Musialski *et al.* (2013) provide a survey of urban reconstruction approaches by following an output-based ordering, namely buildings and semantics, facades and images and finally blocks and cities. As pointed out by Musialski *et al.* (2013), there is a variety of sources for obtaining input data for urban reconstruction. While sensors for aerial and satellite imagery and Light Detection and Ranging (LiDAR) become more easily available, ground-level imagery is still the most prevalent one due to easy acquisition, storage and exchange with lower costs. A common approach used for image-based reconstruction of detailed surfaces is to perform pairwise dense matching of any two registered views and then to combine the computed depth maps, i.e. the depth map fusion

as explained next.

Depth Map Fusion: Fusing depth maps from stereo sequences is a suitable approach for urban reconstruction due to its flexibility and scalability to large amounts of data. In these approaches, the goal is to fuse locally overlapping depth maps rather than producing full 3D models. In a typical approach, there are two steps: first, depth map computation by performing pairwise dense matching between the two views and second, combining the computed depth maps. The fusion can be performed by modeling visibility relations or using a variational or probabilistic approach.

The fusion algorithm used in Pollefeys (2008) is proposed by Merrell *et al.* (2007). Following the depth estimation from plane-sweeping stereo, the fusion is performed by defining a stability measure for each point based on visibility relations to handle occlusions and free space violations. Zach *et al.* (2007b) propose a robust volumetric integration approach for handling outliers using TV regularization and an L_1 data term. Zach (2008) further accelerate the fusion step by modifying the variational formulation to enable paralelized processing on GPU. Zhang *et al.* (2009) propose an iterative method for optimizing all depth maps using loopy belief propagation. Instead of defining explicit visibility constraints as in Merrell *et al.* (2007), occlusions and outliers are handled by the geometric coherence and the color similarity constraints as well as the regularization of belief propagation. Unger (2013) also propose a probabilistic method for fusing depth maps by estimating a probability density function from all the available depth maps. In the end, the most probable estimate is selected from the probability distribution. Fuhrmann and Goesele (2011) propose a method for integrating depth maps of multiple scales. Fusion is performed by computing a hierarchical signed distance function represented in an octree by incrementally adding triangulated depth maps.

Planarity: Similar to binocular stereo, multi-view stereo fails on surfaces where matching cannot be performed reliably. Matching problems constitute a major issue for urban reconstruction due to the large, flat, textureless surfaces frequently observed in urban areas. MVS methods often exploit structural regularities such as planarity and orthogonality to improve reconstruction in those cases. A common approach is to introduce priors on the structure to interpolate smoothly by filling the missing parts and suppressing the noise.

The extraction of detailed 3D information from video streams incurs high computational cost for reconstruction algorithms. By keeping the necessary level of detail low, Cornelis *et al.* (2008) focus on creating compact, memory efficient 3D city models based on simplified geometry assumptions, namely ruled surfaces for facade and road surfaces. However, ruled surfaces are not suitable for modeling some planar surfaces such as vertical facade indentations. Micusik and Kosecka (2009) present a method to overcome these difficulties by exploiting image segmentation cues as well as the presence of dominant scene orientations and piecewise planar structures. In particular, they adopt

a superpixel-based dense stereo reconstruction method by using the Manhattan world assumption with three orthogonal plane normals in the MRF formulation. Similarly, Furukawa *et al.* (2009) also adapt the Manhattan world assumption to improve upon PMVS result (Furukawa and Ponce, 2010). Starting from the 3D point cloud, the dominant plane directions are extracted and clustered to generate plane hypotheses which are then assigned to pixels in a MRF. Sinha *et al.* (2009) employ a general piecewise planar model by computing 3D plane candidates and reconstructing 3D lines based on a sparse point cloud. A piecewise planar depth map is recovered for each image by solving a MRF using graph-cuts. As pointed out by Gallup *et al.* (2010b), non-planar surfaces lead to staircase artifacts or are erroneously flattened to nearby planes when enforcing piecewise planarity. Gallup *et al.* (2010b) first train a classifier to segment an image into piecewise planar and non-planar regions and then enforce a piecewise planarity prior only for planar regions. Non-planar regions are modeled by the output of a standard multi-view stereo algorithm. Gallup *et al.* (2010a) propose an n-layer height map with each layer representing a surface between full and empty space. The height map representation enforces vertical surfaces with no holes.

Primitives: Another way of exploiting piecewise planar structures and shape repetition is to use primitives such as planes, spheres, cylinders, cones and tori. The use of 3D geometric primitives is very common in 3D reconstruction, particularly when reconstructing urban areas. Atomic regions which are geometrically meaningful allow the shape of urban objects to be better preserved. In addition, simplified geometric assumptions can provide significant speedups as well as more compact models.

Lafarge *et al.* (2010) use a library of 3D blocks for reconstructing buildings with different roof forms. Labatut *et al.* (2009); Lafarge and Mallet (2012); Lafarge *et al.* (2013) use 3D-primitives for describing regular structures of the scene. Labatut *et al.* (2009) propose a hierarchical model which robustly detects such simple shapes in the dense point cloud created from sparse depth maps (Labatut *et al.*, 2007). In the end, an energy is minimized to obtain a compact model of the scene using the detected shapes as well as the original points from the Delaunay triangulation. Similarly, Schnabel *et al.* (2007) detect basic shapes in unorganized point clouds using RANSAC and Xiao and Furukawa (2014) use cuboids as volumetric primitives for indoor reconstruction. Primitive arrangement-based approaches provide compactness and reduce complexity. However, they remain simplistic representations and fail to model fine details and irregular shapes. Therefore, Lafarge *et al.* (2013) propose a hybrid approach which is both compact and detailed. Starting from an initial mesh-based reconstruction, they use primitives for regular structures such as columns and walls, while irregular elements are still described by meshes for preserving details. Instead of higher degree primitives, Chauve *et al.* (2010) directly compute an optimized polyhedral reconstruction to obtain an adaptive decomposition of scene by planar primitives.

Omnidirectional Vision

An omnidirectional camera with a 360-degree field of view provides enhanced coverage by eliminating the need for more cameras or mechanically turnable cameras. There are different types of omnidirectional cameras with a visual field that covers a hemisphere or even approximately the entire sphere. Catadioptric cameras combine a standard camera with a shaped mirror, such as a parabolic, hyperbolic, or elliptical mirror while dioptric cameras use purely dioptric fisheye lenses.

One classification often used in the literature for omnidirectional cameras is based on the projection center: central and non-central. In central cameras, the optical rays to the viewed objects intersect in a single point in 3D which is known as the single effective viewpoint property. This property allows the generation of geometrically correct perspective images from the images captured by omnidirectional cameras and consequently, application of epipolar geometry which holds for any central camera.

In contrast to pinhole cameras, calibration of omnidirectional cameras cannot be modeled by a linear projection due to very high distortion. The model should take into account the reflection of the mirror in the case of a catadioptric camera or the refraction caused by the lens in the case of a fisheye camera. Geyer and Daniilidis (2000) provide a unifying theory for all central catadioptric systems which is known as unified projection model in the literature and widely used by different calibration toolboxes (Mei and Rives, 2007; Heng *et al.*, 2013, 2015). They prove that every projection, both standard perspective and catadioptric using a hyperbolic, parabolic, or elliptical mirror, can be modeled with projective mappings from the sphere to a plane where the projection center is on a sphere diameter and the plane perpendicular to it. Mei and Rives (2007) improve upon the unified projection model of Geyer and Daniilidis (2000) to account for real-world errors by modeling distortions with well identified parameters.

Applications: For feature based applications such as navigation, motion estimation and mapping, the large field of view enables extraction and matching of interesting points from all around the car. For instance, omnidirectional feature correspondences improve the rotation estimate significantly when doing visual odometry or SLAM (Scaramuzza and Siegwart, 2008). 3D perception also benefits from the unified view offered by omnidirectional sensors, despite the limited effective resolution which leads to noisy reconstructions. Alternative laser-based solutions provide only sparse point clouds without color, are extremely expensive and suffer from rolling shutter effects. Schönbein and Geiger (2014) propose a method for 3D reconstruction through joint optimization of disparity estimates from two temporally and two spatially adjacent omnidirectional views in a unified omnidirectional space by using plane-based priors. Häne *et al.* (2014b) extend the plane-sweeping stereo matching for fisheye cameras by incorporating the unified projection model for fisheye cameras directly into the plane-sweeping stereo matching algorithm.

2.3 Optical Flow Estimation

Optical flow is the problem of estimating the motion of pixels between consecutive frames in time. In the traditional sense, the estimation is based on minimizing the brightness or color difference between corresponding pixels on two frames, which is known as the brightness constancy assumption. More specifically, optical flow algorithms estimate a 2D flow vector at each pixel which encodes the horizontal and vertical motion of the pixel from one frame to the next. In the literature, the first frame is often referred as the reference frame and the second frame as the target frame. Since the number of variables to estimate is twice the number of observations, i.e. pixels[4], optical flow is an under-constrained problem. Note that optical flow is purely based on the motion of pixels on the image plane rather than the 3D motion of the objects in the scene, i.e. scene flow.

Optical flow has a variety of applications related to the processing of video sequences such as video denoising, frame interpolation and video summarization. Since optical flow is concerned with the motion of pixels on the image plane, it is related to the structure of the objects and the scene as well as the motion of the observer and the objects. In that sense, there are many related problems in robotics, especially in the autonomous driving context including ego-motion estimation, structure from motion and object detection and tracking.

2.3.1 Traditional Approaches

Due to the under-constrained nature of the problem, optical flow algorithms introduce additional constraints to estimate the flow fields. Two classical approaches are the patch-based approach by combining information over a neighborhood, and the use of the smoothness assumptions on the flow fields as a global regularization.

Local Least-Squares Estimation: In the first approach, optical flow is formulated as the minimization of the sum-of-squared differences (SSD) of image intensities over a small window with respect to flow displacements. Lucas and Kanade (Lucas and Kanade, 1981), the seminal method for patch-based approaches, is based on the linearization of the SSD displacement function by performing a first order Taylor series expansion:

$$E_{SSD}(u,v) = \sum_{(x,y)\in R} (I(x+u,y+v,t+1) - I(x,y,t))^2 \tag{2.6}$$

$$\approx \sum_{(x,y)\in R} (u \cdot I_x(x,y,t)) + v \cdot I_y(x,y,t) + I_t(x,y,t))^2 \tag{2.7}$$

[4]Even in case of RGB images, the intensity values of each channel are highly correlated.

The linearization leads to optical flow constraint equation which is also known as the linearized brightness constancy constraint:

$$u \cdot I_x + v \cdot I_y + I_t = 0 \tag{2.8}$$

where subscripts denote partial derivatives. The approximation only holds for small motions due to the linearization.

The ill-posed problem is addressed by combining multiple constraints in a region on the image. The usual approach for minimizing the approximated objective is gradient descent. Optimization using gradient descent can be performed in different ways, either as an additive increment to the parameters (Lucas and Kanade, 1981) or as an incremental warp that is composed with the current estimate of the warp (Shum and Szeliski, 2000). In each iteration, various gradient descent approximations can be used such as Gauss-Newton, Newton, diagonal Hessian, Levenberg-Marquardt, and the steepest descent (Baker and Matthews, 2004).

Similar to local and global approaches in stereo, Lucas and Kanade is a local method for optical flow since each patch is considered independently. The spatial coherence is enforced to some extent, since the flow within an image window is assumed to be constant. However, deciding the window size remains a problem. While small windows are likely to produce wrong matches due to ambiguities, large windows often result in violations of the constant flow assumption within the window.

Global Formulation: Instead of solving for each pixel's motion independently, Horn and Schunck formulates optical flow estimation as the minimization of a global energy function defined over all flow vectors (Horn and Schunck, 1981). While Lucas and Kanade is a local least squares solution, Horn and Schunck enforces a global regularization on the flow fields. The image is considered as a function of continuous variables in spatial and temporal domain. The energy function is typically composed of a data term to penalize the brightness change between corresponding pixels and a regularizer to penalize local change in flow:

$$\begin{aligned} E(u,v) \;=\; & \int\int (I(x+u(x,y),\, y+v(x,y),\, t+1) - I(x,y,t))^2 + \\ & \lambda \cdot (\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2)\; dx\, dy \end{aligned} \tag{2.9}$$

The energy as initially proposed by Horn and Schunck (1981) uses a quadratic penalty on the optical flow constraint and the gradient of the flow field. Minimizing this highly non-convex energy function with respect to continuous flow fields is very prone to local optima. Similar to Lucas and Kanade, the brightness constancy assumption can be linearized to obtain a convex energy with a unique optimum. Then, the linearized energy can be minimized by discretizing it spatially and performing gradient descent on the discretized objective. The resulting linear system of equations are typically solved using

iterative techniques like Gauss-Seidel or Successive Over-relaxation (SOR).

One challenge for the global methods is to find a compromise between resolving ambiguities and avoiding very smooth estimates. While increasing regularization, i.e. larger values of λ in Eq. 2.10, helps with missing or noisy local cues, it also results in over-smoothing flow discontinuities. This is due to the quadratic penalty in the original formulation of Horn and Schunck for enforcing the brightness constancy assumption and smooth flow fields. These assumptions are often violated due to varying illumination conditions, occlusions and discontinuities. Using robust penalty functions is a generic way of alleviating these problems. Instead of quadratic functions which correspond to a Gaussian prior on local changes, Black and Anandan (1993) propose to use more heavy-tailed functions such as a Student-t distribution, i.e. Lorentzian penalty which can better handle the violations by penalizing large changes less.

Another challenge is large displacements since the global formulation is also limited to small motions due to linearization. In addition to non-linearized solutions (Nagel, 1987; Alvarez *et al.*, 2000), the coarse-to-fine strategy is a common way of dealing with large displacements (Anandan, 1989; Black and Anandan, 1996; Brox *et al.*, 2004). The idea is to create a pyramid of coarse-to-fine images by down-sampling the original image at each level. The estimate at a coarser level is used to initialize the estimate at the next finer level. This way, only small flow increments need to be estimated. The optical flow estimate is iteratively refined at each level until the full resolution. This idea has been recently utilized in deep learning methods using a differentiable warping operation (Ilg *et al.*, 2016; Ranjan and Black, 2016).

Over the years, models of various forms have been proposed by combining different data and regularization terms. Vogel *et al.* (2013a) systematically evaluates pixel- and patch-based data costs in a unified testbed on the KITTI dataset (Geiger *et al.*, 2012). On real data, patch-based terms are found to perform better than pixel-based terms. Zach *et al.* (2007a) propose to replace the quadratic penalization with the L_1 norm in the data term and the Total Variation regularization in order to better preserve discontinuities in the flow field. However, Total Variation favors fronto-parallel surfaces which is often violated in real-world scenes. Bredies *et al.* (2010) propose second-order Total Generalized Variation (TGV) model as a higher-order regularization. TGV priors can better represent real data as they leverage a piecewise affine motion model. Ranftl *et al.* (2014) propose a non-local extension of TGV with a larger support size and the possibility to enforce additional prior knowledge using the support weights to incorporate soft-segmentation cues into the regularization term. Zimmer *et al.* (2011) provide a detailed comparison of image- and flow-driven regularizers for the variational formulation and discuss the qualities of different data terms. In addition to the model, the optimization method and its implementation also affect the performance of variational methods. Sun *et al.* (2014) investigate the success of various optical flow methods and propose an approach for optimizing a classical formulation with modern techniques.

Sparse Matches

In variational methods, a typical way of handling large displacements is the coarse-to-fine strategy. While this strategy works for large structures of low complexity, fine geometric details are often lost in the process. These problems can be alleviated by integrating sparse features into the variational formulation as proposed by Brox and Malik (2011). The feature matches obtained from the nearest neighbor search on a coarse grid are used as a soft constraint in the coarse-to-fine optimization. Another sparse matching method for large displacements is Epic Flow (Revaud *et al.*, 2015). In Epic Flow, the coarse-to-fine strategy is replaced by an interpolation of sparse matches to initialize a dense optimization at full resolution. Sparse matches are obtained using Deep Matching (Weinzaepfel *et al.*, 2013).

Discrete Optical Flow: In contrast to the traditional formulation of optical flow as a continuous optimization problem, recent approaches leverage discrete optimization strategies in order to overcome local minima and gain robustness. These approaches can be considered as sparse matching, since a set of match hypotheses are created and further refined in a discrete optimization framework. Menze *et al.* (2015) use appearance features and approximate nearest neighbor search to establish a set of proposals as candidates to be used as input to a 4-connected MRF with pairwise smoothness constraints. Inference is made feasible by restricting the number of matches to the most likely ones with non-maxima suppression and exploiting the truncated form of the pairwise potentials. Chen and Koltun (2016) argue that the heuristic pruning used to make inference feasible destroys the highly regular structure of the space of mappings and propose a discrete optimization over the full space. Min-convolutions are used to reduce the complexity and to effectively optimize the large label space using a modified version of tree-reweighted message passing algorithm by Kolmogorov (2006).

Motion Models

Instead of a 2D vector representation at each pixel, flow fields in a region can be defined by a parametric model with an associated parameter vector. Parametric motion models present a compact representation for approximating the true motion of a region with a coherency assumption on the motion of pixels inside the region. Simple polynomial motions such as the translation or the affine motion model can be used to represent the motion in small regions. The translation model only holds for very small regions due to its restrictiveness. The affine motion model originates from the assumption that rigidly moving planar surfaces in the world are projected orthogonally on the image plane. More complex motion models that allow deviations from restrictive assumptions are required for modeling motions in larger regions which do not adhere to the assumptions on planarity or orthographic projection.

In optical flow, most parametric models rely on simple motion models such as polynomial ones for accurate approximations of motion in local, piecewise regions. The piecewise parametric models assume a low-order parametric motion models within each segment (Black and Anandan, 1996; Black and Jepson, 1996; Ju *et al.*, 1996). One common source of error for these models is when image segments do not correspond to motion segments. Therefore, Yang and Li (2015) model flow fields by multiple parametric motion models in a piecewise manner by allowing the size and shape of each piece to change adaptively. The number of pieces and the associated motion model for each piece are estimated in a combined discrete and continuous optimization framework.

Layered Motion Models: Layered models take advantage of the fact that motion and scene geometry contain important information about each other. A common way is to alternate optimization between the segmentation of the scene and motion estimation (Schoenemann and Cremers, 2008). An alternative approach is to jointly reason about the segmentation and the motion by assuming a parametric motion model for each segment (Cremers and Soatto, 2005). A coupled approach can avoid local optima by improving in occlusions or in cases when an object appears in the wrong layer but with the correct motion. Unger *et al.* (2012) propose a variational formulation for joint motion estimation and segmentation with explicit occlusion handling. Sun *et al.* (2010) propose a generative layered model that combines mixture models with an image segmentation model. Sun *et al.* (2012) replace gradient-based inference algorithm of Sun *et al.* (2010) with a discrete layered model based on a sequence of ordered Markov Random Fields. Instead of a locally connected MRF model, Sun *et al.* (2013) formulate a fully-connected layered model (Krähenbühl and Koltun, 2011) that enables global reasoning in order to capture long-range interactions and improve segmentation.

Learned Subspace Models: In learned subspace models, optical flow field $\mathbf{f}$ is represented as a weighted sum of N basis flow fields:

$$\mathbf{f} = \sum_{n=1}^{N} w_n \mathbf{b}_n \tag{2.10}$$

First, a set of dense motion fields is computed from a set of training videos. Then, principal components of the resulting flow fields are extracted and used as the basis flow fields. In this context, optical flow estimation is reduced to the estimation of the linear coefficients w_n for each basis vector. An early approach from Black and Yacoob (1997) proposes a method for recognizing human facial expressions from the coefficients of a parameterized motion model where the relative motion of eyebrows and mouth are modeled using a hand-coded model. Fleet *et al.* (2000) propose to learn the basis flow fields for the motion of human mouths from example patches. The coefficients are estimated using a robust estimator with a generic smoothness model in a warping-based optimiza-

tion scheme. A more recent approach called PCA Flow proposes to obtain dense optical flow for the full image from sparse matches Wulff and Black (2015). The optical flow is represented as a weighted sum of basis flow fields learned from reference flow fields computed from Hollywood movies. During inference, optical flow is estimated by finding the weights which minimize the error with respect to the detected sparse feature correspondences. This approach results in a very fast algorithm but produces overly smooth flow fields. Therefore, a slower layered approach where each layer is modeled with PCA Flow is proposed to better handle flow discontinuities.

Epipolar Flow and Semantics

In some special cases such as in the context of autonomous driving, simplifying assumptions can be leveraged for estimating optical flow. The assumption of a static scene or the decomposition of a scene into rigidly moving objects reduces optical flow to a matching problem along epipolar lines radiating from the focus of expansion. Yamaguchi *et al.* (2013) propose a slanted-plane Markov random field that represents the epipolar flow of each segment with slanted planes. Yamaguchi *et al.* (2014) extend this to a joint stereo and flow formulation by removing time consuming optimization part and proposing a new semi global block matching algorithm. In contrast to these approaches, Bai *et al.* (2016) use the slanted plane model only for background flow estimation. Using an instance segmentation, inference is formulated as an independent epipolar flow estimation problem for each moving object. This is particularly useful for dynamic scenes with many independently moving objects as in KITTI 2015 dataset.

Optical flow estimation can benefit from the available domain knowledge, for example by exploiting semantics either from a generic segmentation of the scene or from specific instances as mentioned above. For scenes composed of a static background and dynamic moving objects, Bai *et al.* (2016) extract the traffic participants using instance-level segmentation and estimate the optical flow independently for different instances. Sevilla-Lara *et al.* (2016) use semantic segmentation for optical flow estimation to improve the flow estimation near object boundaries as well as to reason about spatial relationships between objects for depth ordering in layered models. In addition, semantic information can be further exploited by using different motion models for each object type. The motion in planar regions such as roads is modeled by estimating homographies, whereas independently moving objects, i.e. cars, are modeled via affine motion models to allow for deviations. Complex objects like plants are modeled with a spatially varying dense flow field, i.e. Discrete Flow (Menze *et al.*, 2015). Finally, the constancy of object identities over time can be used to encourage temporal consistency of the optical flow.

Deep Learning

Due to their success in many vision problems, convolutional neural networks have also been utilized for the optical flow estimation. There are two main lines of work in deep

learning for optical flow: end-to-end learning of flow from the full image and learning similarity measure on patches, typically using Siamese networks. Here, we focus on the first line of work and leave the discussion of patch-based networks to Section 3.1. End-to-end approaches use a deep convolutional neural network which takes as input two images and directly outputs a flow field. One difficulty is the model's high capacity and the associated large amount of data required to train it.

The most representative work for learning optical flow using a CNN is FlowNet (Dosovitskiy *et al.*, 2015). FlowNet consists of a contracting part which acts like an encoder and an expanding part with deconvolutions to produce the flow result, i.e. the decoder. In the paper, two types of architectures are explored: the simple network explained and a more complex network with a layer that correlates feature vectors at different image locations. One problem in learning optical flow is the limited amount of training data. KITTI 2012 (Geiger *et al.*, 2012) and KITTI 2015 (Menze and Geiger, 2015) only provide around 200 training examples each while Sintel (Butler *et al.*, 2012) has 1041 training image pairs. Since these datasets are too small to train large CNNs, a synthetic dataset called the Flying Chairs was created by rendering 3D chair models on top of images from Flickr. FlowNet shows decent performance while being much faster than many flow algorithms but cannot reach state-of-the art performance on benchmarks.

Inspired by the coarse-to-fine methods in traditional optical flow estimation, Ranjan and Black (2016) propose SPyNet, an architecture based on warping. Each layer of the network represents a different scale and only estimates the residual flow with respect to the warped image. SPyNet achieves similar performance as FlowNet with a faster and much smaller network which opens the way for its deployment in embedded systems. Another method which uses warping is FlowNet2 proposed by Ilg *et al.* (2016). FlowNet2 improves upon FlowNet by stacking architectures and fusing the stacked network with a subnetwork specialized to small motions. Similar to SPyNet, FlowNet2 also inputs the warped image into the stacked networks. However, each stacked network estimates the flow between the original frames instead of the residual flow. In addition to the Flying Chairs dataset, another synthetic dataset called the FlyingThings3D is used in combination for exploring different strategies in presenting the data to the network (Mayer *et al.*, 2016). FlowNet2 performs on par with state-of-the-art methods on Sintel and outperforms other methods on KITTI 2015 while being one of the fastest. Different network variants are provided for the spectrum between 8 fps and 140 fps, allowing trade-off accuracy and computational resources.

2.3.2 Datasets

In this section, we introduce the popular datasets for optical flow estimation together with the common evaluation criteria as well as ways of visualizing and comparing results. To evaluate the performance of methods, three measures commonly used are average angular error (AAE), average endpoint error (EPE) and the outlier ratio based on a threshold. In the qualitative evaluations, there are two main visualization techniques commonly

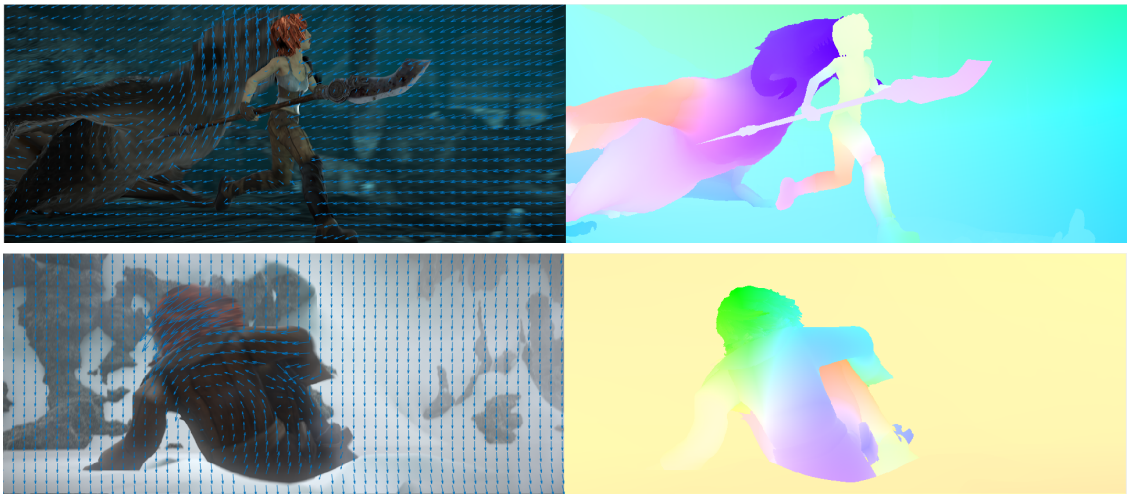


Figure 2.3: **Flow Visualization.** This figure illustrates two visualization techniques used for flow on two examples from the MPI Sintel dataset (Butler *et al.*, 2012): sub-sampled arrow visualization of motion vectors overlaid on the image (**left**) and dense false coloring where color hue defines the direction and saturation shows the magnitude of the motion vector (**right**).

used: sub-sampled arrow visualization of motion vectors and dense false coloring where color hue defines the direction and saturation shows the magnitude of the motion vector (see Fig. 2.3).

One of the first publicly available benchmarks proposed for optical flow is the Middlebury benchmark [5] by Baker *et al.* (2011). The Middleburry benchmark consists of eight synthetic sequences with eight frames each. Ground-truth is obtained for one pair of frames per sequence by tracking hidden fluorescent textures with UV light. The benchmark is the first to provide sequences with dense and accurate ground-truth up to subpixel precision. However, it lacks the complex structures, lighting variation and shadows which occur in real-world scenes. In addition, it comprises very small motions of up to twelve pixels and it is very limited in terms of number of images in the dataset. The methods submitted are evaluated based on AAE and EPE.

Currently, Sintel (Butler *et al.*, 2012) and KITTI (Geiger *et al.*, 2012, 2013) are the two most popular datasets for the evaluation of optical flow algorithms. The MPI Sintel Flow benchmark[6] by Butler *et al.* (2012) uses an open source movie Sintel, an animated short film, to render scenes of varying complexity with optical flow ground-truth. In total, Sintel comprises 1,628 frames each in clean and final passes with varying complexity. While the clean pass introduces illumination of various kinds, the final pass adds atmospheric effects, blur, color correction and vignetting. In addition to the average end-

[5] `http://vision.middlebury.edu/flow/`
[6] `http://sintel.is.tue.mpg.de/`

point error, the benchmark website provides different rankings of the methods based on velocity, occlusion boundaries and dis-occlusions. Compared to Middlebury, it contains larger motions, many non-rigidly moving objects and image degradations such as motion and defocus blur in the final pass.

Despite the new challenges introduced, the Sintel benchmark is still a synthetic dataset. As further explained in Section 2.1.3, the KITTI 2012 and 2015 datasets provide realistic scenarios in autonomous driving context (Geiger *et al.*, 2012; Menze and Geiger, 2015). Considering data hungry deep learning methods, both KITTI and Sintel datasets are still too limited in size. To train convolutional neural networks, synthetic datasets have been used as a solution by rendering 3D models of chairs or other objects on top of random background images (Dosovitskiy *et al.*, 2015; Mayer *et al.*, 2016). Producing large amount of realistic data to train deep neural networks is one of the remaining challenges in optical flow estimation.

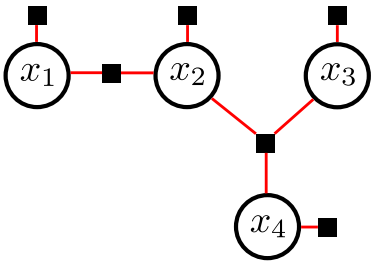
2.4 Inference Techniques

In computer vision, we build models to relate image observations to various kinds of information that we want to estimate from the image. The model is typically associated with a set of variables. The observation variables are measured image pixels or a group of pixels and the output variables represent the quantities that we want to estimate such as depth or semantic information about the image. We often use probabilistic models in order to take into account uncertainty and deal with imperfect observations. A typical way of representing probability models is graphical models. Based on the defined random variables, a graphical model describes how variables interact using structural assumptions. The goal is to estimate output variables or answer questions related to these variables, i.e. inference.

Graphical models are represented in terms of a graph which encodes the dependency structure of the problem and defines constraints between the random variables. Probabilistic graphical models encode a joint or conditional probability distribution over all feasible solutions given some observations. There are different types of graphical models, typically defined by the graph structure. Each graph encodes some conditional independence assumptions which can be defined over discrete or continuous variables or even over a mixture of these. Directed graphical models, also known as Bayesian networks, are represented as a directed acyclic graph which encodes conditional probability distributions. Undirected graphical models, also known as Markov random fields (MRF) define a family of joint probability distributions using an undirected graph with a set of cliques. Markov conditions mean that the distribution can be expressed as a product of potentials defined on the cliques of the graph, i.e. factorization. In undirected graphical models, it might become cumbersome to define pairwise interactions between all pairs of variables. A special form, called factor graphs, provides a more convenient way with factor nodes in addition to variable nodes to make the factorization explicit. In most cases, we have access to some measurements which correspond to observations in the graphical model. In that case, conditional distributions can be expressed using conditional random fields (CRF).

The factor graph provides a factorization which defines a probability distribution as the multiplication of all the factors divided by a normalization factor (Fig. 2.4). We define an energy function for each factor and formulate the problem as an energy minimization over all possible solutions. Given a factor graph, we can use the model for inference after learning its parameters. In computer vision, a frequent form of inference is maximum a posteriori (MAP). In MAP inference, the goal is to find the state of maximum probability which corresponds to the the optimal prediction according to an expected loss function. We are also often interested in marginal probabilities for each factor, i.e. probabilistic inference.

While inference in tree-structured graphs can be solved exactly, approximate inference methods are required for graphs with cycles. Due to frequently used pairwise factors on grid-like graphs which correspond to pixels on the image, computer vision tasks are typ-



$$p(x_1,x_2,x_3,x_4) = \frac{1}{Z}\phi(x_1)\phi(x_2)\psi(x_1,x_2) \\ \phi(x_3)\phi(x_4)\psi(x_2,x_3,x_4)$$

Figure 2.4: **Factor Graph.** This figure shows an example of a factor graph defined over x_1, x_2, x_3, x_4 with a unary potential for each, one pairwise potential and one triplet potential. The corresponding factorization $p(x_1, x_2, x_3, x_4)$ is shown on the right as defined in Eq. 2.11. Note that subscripts of different potentials are dropped for clarity.

ically loopy by nature, i.e. the corresponding graph contains cycles. Therefore, using approximate inference algorithms is a common way of solving vision problems. There are three major approaches to approximate inference: message passing via Loopy Belief Propagation, inference as optimization as in the Structured Mean Field approach, and sampling-based inference such as particle filtering, Gibbs sampling and MCMC sampling. The first two are deterministic inference techniques while sampling approaches are stochastic approximations that randomize to approximate expectations. Sampling approaches can be more exact with approximation guarantees, but they are usually slower than deterministic variants.

In this thesis, we use a standard parallel implementation of Loopy Belief Propagation for discrete inference in optical flow estimation. For inferring continuous plane parameters in stereo estimation, we use Particle Belief Propagation with TRW-S in the inner loop iterations. Furthermore, we use MCMC for sampling object disparity proposals in stereo estimation. In the following, we briefly explain each of these approaches. Please refer to the references and the related textbooks for a detailed description of these methods as well as theorems and proofs (Andrieu *et al.*, 2003; Bishop, 2006; Wainwright and Jordan, 2008; Brooks *et al.*, 2011; Nowozin *et al.*, 2011; Barber, 2012).

Notation

Let $p(x_1, \cdots, x_n)$ denote a joint probability distribution over the set of n discrete random variables $x_1, \cdots, x_n$. A graphical model specifies a factorization of this probability into a product of non-negative potential functions, each defined over a small number of variables, i.e. factors:

$$p(x_1, \cdots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} \phi_i(x_i) \prod_{\alpha=1}^{m} \psi_\alpha(\mathbf{x}_\alpha) \tag{2.11}$$

$\phi_i(x_i)$ is typically referred as the unary term and represent the local information over the states of x_i. The other m functions $\psi_\alpha(\mathbf{x}_\alpha)$ are defined over a subset of variables $\mathbf{x}_\alpha$. In factor graphs, each random variable x_i is represented by a circle node and each factor corresponding to a potential is represented by a filled square. If $x_i \in \mathbf{x}_\alpha$, then there is an edge connecting the random variable x_i to the factor which corresponds to $\mathbf{x}_\alpha$ on the factor graph. The normalization constant Z is often referred as the partition function.

2.4.1 Belief Propagation (BP)

A popular algorithm for inference in graphical models is the belief propagation (BP) algorithm. The belief propagation algorithm can be used to perform MAP inference, i.e. max-product BP, or to compute marginal distributions for all variables and factors, i.e. sum-product algorithm. MAP inference is finding the state for each x_i which corresponds to the maximal value of the joint probability $p(x_1, \cdots, x_n)$:

$$\mathbf{x}^* = \underset{x_1,\cdots,x_n}{\operatorname{argmax}} \; p(x_1, \cdots, x_n) \tag{2.12}$$

The marginal probabilities over a random variable or a set of random variables are defined as follows:

$$p(x_i) = \sum_{\mathbf{x}\setminus x_i} p(\mathbf{x}) \tag{2.13}$$

$$p(\mathbf{x}_\alpha) = \sum_{\mathbf{x}\setminus \mathbf{x}_\alpha} p(\mathbf{x}) \tag{2.14}$$

The algorithm works by defining messages between variables and factors. The message $\mu_{\alpha\to i}$ from factor α to variable i, and the opposite direction $\mu_{i\to\alpha}$ is a vector over the states of x_i. In the sum-product algorithm, the messages are defined recursively as

$$\mu_{\alpha\to i} = \sum_{\mathbf{x}_\alpha\setminus x_i} \psi_\alpha(\mathbf{x}_\alpha) \prod_{j\in N(\alpha)\setminus i} \mu_{j\to\alpha}(x_j) \tag{2.15}$$

$$\mu_{i\to\alpha} \propto \phi_i(x_i) \prod_{\beta\in N(i)\setminus\alpha} \mu_{\beta\to i}(x_i) \tag{2.16}$$

where $\propto$ indicates a possible normalization over the vector and $N(\cdot)$ defines the neighborhood of a factor or a variable. $N(i)$ stands for all factors that are neighbors of variable node i, i.e. all α for which $x_i \in \mathbf{x}_\alpha$ and $N(\alpha)$ for all variables that are neighbors of factor α. In case of max-product, the messages from factors to variables are changed as follows:

$$\mu_{\alpha\to i} = \max_{\mathbf{x}_\alpha\setminus x_i} \left\{ \psi_\alpha(\mathbf{x}_\alpha) \prod_{j\in N(\alpha)\setminus i} \mu_{j\to\alpha}(x_j) \right\} \tag{2.17}$$

After initialization, messages are sent in iterations between factors and variables which is known as message passing. Upon convergence or after a number of iterations, marginal probabilities $p(x_i)$ and $p(\mathbf{x}_\alpha)$ can be approximated in terms of beliefs:

$$b_i(x_i) \propto \phi_i(x_i) \prod_{\alpha \in N(i)} \mu_{\alpha \to i}(x_i) \tag{2.18}$$

$$b_\alpha(\mathbf{x}_\alpha) \propto \psi_\alpha(\mathbf{x}_\alpha) \prod_{j \in N(\alpha)} \mu_{j \to \alpha}(x_j) \tag{2.19}$$

For tree-structured graphs without cycles, beliefs correspond to the marginal probabilities. When the graph contains cycles, beliefs provide only an approximation to the marginal probabilities. In case of max-product, the approximate MAP solution is defined as the highest belief state $b_i(x_i)$ for each x_i. Since the messages are still well-defined on a loopy graph, we can apply the BP algorithm for a number of iterations, i.e. loopy BP. In practice, the algorithm does not always converge, resulting in poor approximations to the true marginals. When it converges, the approximate inference still leads to very good results for a wide range of applications.

Since products of small probabilities result in numerical instabilities, a common practice is to use negative log probabilities. Then, the max-product becomes a min-sum algorithm. When using negative log probabilities, all entries are initialized to zero. The resulting log-sum-exp expression with large numbers might still lead to numerical instabilities. As a solution, the log-sum-exp trick can be used:

$$\log \sum_i \exp(x_i) = a + \log \sum_i \exp(x_i - a) \tag{2.20}$$

where $a = \max_i \ x_i$.

Felzenszwalb and Huttenlocher (2006) propose different techniques for speeding up the standard belief propagation algorithm. In case of the discontinuity costs commonly used in early vision, each message update can be expressed as a min-convolution which can be computed in linear time using the fast Fourier transform. Furthermore, BP can be performed more efficiently for bipartite graphs[7] such as the grid-structured graphs. Since messages sent from one set do not depend on the messages sent from the other set, we can use a parallel implementation.

Approximate Inference using Variational Principles

As the loopy belief propagation performs well in practice, but without any convergence guarantees, there has been a significant effort to understand its behavior and find a convergent message passing algorithm. A common approach is the variational interpretation of message-passing algorithms, starting from the KL-divergence between the input dis-

[7] A bipartite graph is one where the nodes can be split into two sets so that every edge connects pairs of nodes in different sets (Felzenszwalb and Huttenlocher, 2006).

tribution and its factorization. Yedidia *et al.* (2005) show the close connection between the loopy BP and a constrained energy function called the Bethe free energy in physics. If the algorithm converges, then it converges to a fixed point of the Bethe free energy. This connection allows the investigation of the stationary points of the Bethe free energy to find convergence guarantees using more general optimization techniques than BP.

One line of work in variational methods is based on a tractable approximation to the free energy by approximating the entropy term. In other words, the sum-product updates are formulated in terms of the Bethe approximation which replaces the entropy term in the free energy by the Bethe entropy. The details of this process are explained in Wainwright and Jordan (2008). For factor graphs without cycles, the Bethe energy is convex over the set of constraints and BP is exact. However, for the case with cycles, the Bethe energy is non-convex and the BP algorithms may fail to converge.

Tree-reweighted Message Passing: There are methods which offer generalizations of the Bethe approximation for handling the non-convexity. The goal is to find a generalization which leads to a message-passing algorithm with convergence guarantees. One approach introduces a wider class of functions, convex free energies, which are convex over the set of constraints for any factor graph. Following that direction, the Tree-reweighted (TRW) free energy is inspired by the idea of maximizing a concave lower bound on the energy (Wainwright and Jordan, 2008). The TRW energy consists of a linear combination of free energies defined on spanning trees of the factor graph. If their fixed point satisfies a certain condition, which is defined as the tree agreement, then it is guaranteed to give a MAP solution, i.e. a global minimum of the energy (Wainwright and Jordan, 2008; Kolmogorov, 2006). This can be viewed as a convexified form of the Bethe approximation with strong convergence guarantees as well as a lower bound on the best achievable energy.

Kolmogorov (2006) show that TRW obtains a lower energy than both graph cuts and the BP for stereo problems with Potts interactions. Then, they modify the TRW algorithm so that the value of the bound is guaranteed not to decrease and find at least a local maximum of the bound. The algorithm is based on sequential scheduling of TRW updates called TRW-S updates. In their experiments, the sequential updates outperform a standard parallel scheduling of the TRW max-product updates.

Continuous State Spaces

In case of continuous random vectors, sums are replaced by integrals. The inference problem is typically harder for the continuous case due to the large number of integrals that need to be computed. In that case, the exact marginals can be computed only for special cases, e.g. a tree with Gaussian potentials. More specifically, a message or belief function that represents a probability is modeled as a Gaussian and only mean and variance values are propagated across the graph. Similar to the discrete case, the BP algorithm may not converge for graphs with cycles. For these situations, there are

various methods that approximate messages for example as Gaussian in an Expectation Propagation algorithm, a mixture of Gaussians in Non-parametric BP (Sudderth *et al.*, 2003), as a weighted set of particles in Particle BP (Ihler and McAllester, 2009) or as some combination of orthogonal functions in Stochastic Message Passing (Noorshams and Wainwright, 2013). Here, we focus on a max product variant of particle BP as it is the one that we use for inferring continuous plane parameters.

Max-product Particle BP: Particle BP approaches investigate generalizations of particle filtering defined on Markov chains to more general graph structures. In these approaches, each distribution is represented using a finite collection of samples, i.e. particles. Two general approaches are to draw a set of particles for each message on the graph or to create a set of particles for each variable. Either approach can be shown to produce the correct answer in the limit as the number of particles goes to infinity (Ihler and McAllester, 2009). However, the computational efficiency is an important issue which forces approaches to consider small number of particles.

In the second approach, the set of particles can be considered as the set of possible values for each variable in a discrete model. Then, the particle max-product algorithm alternates between computing the MAP solution on the discrete problem and sampling new particles based on this solution (Ihler and McAllester, 2009; Peng *et al.*, 2011). The process is repeated in iterations. New particles can be generated in a greedy way by adding Gaussian noise to the MAP solution. There are diverse particle approaches which augment the particle sets at each node with samples from their neighbors (Besse *et al.*, 2014) and by Gaussian random-walk followed by a potential particle selection step (Pacheco *et al.*, 2014).

2.4.2 Markov Chain Monte Carlo (MCMC) Sampling

Monte Carlo methods are numerical techniques for computing approximate estimates using random sampling. Monte Carlo approximation can provide an unbiased estimate of an analytically intractable integral with a finite sum. The goal is to generate a set of samples $x_1, \cdots, x_N$ from a target distribution $p(x)$. Basic Monte Carlo methods such as inverse transform and rejection sampling can be used to sample from simple distributions with low-dimensionality. For more complex distributions, Markov Chain Monte Carlo methods can be used.

Markov Chain: Instead of directly sampling from the target distribution $p(x)$, Markov Chain Monte Carlo methods sample from a Markov chain. In Markov Chains, conditional probability distribution of successor states in the process depends only on the current state:

$$P(x^{(i)}|x^{(i-1)}, \cdots, x^{(1)}) = P(x^{(i)}|x^{(i-1)}) \tag{2.21}$$

where i is the sample index. This is known as *Markov Property* and results in simplified joint distribution:

$$P(x^{(1)}, \cdots, x^{(N)}) = P(x^{(1)}) \prod_{i=2}^{N} P(x^{(i)}|x^{(i-1)}) \tag{2.22}$$

A Markov chain for which the transition operator does not depend on time, i.e. i, is called homogeneous Markov chain. Homogeneous Markov chains can be associated with a transition density or kernel $K(x^{(i)}|x^{(i-1)})$. Given an initial state $x^{(1)}$, we draw $x^{(2)}$, then conditioned on $x^{(2)}$, we draw $x^{(3)}$ and continue iteratively using $K(x^{(i)}|x^{(i-1)})$.

In the discrete case, the transition probabilities can be encoded in a transition matrix $\mathbf{K}$. Then, the probability distribution for the new state $\mathbf{p}_i$ is obtained by multiplying $\mathbf{p}_{i-1}$ by the transition matrix $\mathbf{K}$:

$$\mathbf{p}_i^T = \mathbf{p}_{i-1}^T \mathbf{K} \tag{2.23}$$

where each row in $\mathbf{K}$ sums to 1 to ensure valid probability distributions.

Convergence: An important property for Markov Chains is the stationarity for convergence. A stationary distribution $\mathbf{p}$ satisfies the following:

$$\mathbf{p}^T = \mathbf{p}^T \mathbf{K} \tag{2.24}$$

A key conclusion is that for each ergodonic Markov chain, there exists a unique distribution $\mathbf{p}$ for which $\mathbf{p}^T = \mathbf{p}^T \mathbf{K}$, i.e. Markov Chain converges to a unique stationary distribution. Ergodicity is defined by the irreducibility and the aperiodicity. The irreducibility means that each state can be visited starting from each one in a finite number of steps. In an aperiodic Markov Chain, the occurrence of states is not restricted to periodic events, but any state may occur at any time. Please refer to (Andrieu *et al.*, 2003; Brooks *et al.*, 2011) for corresponding theorems and proofs.

Metropolis-Hastings Sampling

By designing a transition kernel K whose stationary distribution is the target distribution $p(x)$, we can generate samples from the Markov Chain that eventually converges to the target distribution. After ignoring samples from an initial burn in period, the generated samples can be used to estimate properties related to the target distribution.

The goal is to obtain a transition kernel K for a target distribution $p(x)$. Metropolis-Hastings does not explicitly design a transition matrix, but defines a procedure to draw a new sample given the previous one using a proposal distribution which is easy and efficient to sample from. Given the proposal distribution $q(x'|x)$ and the current sample

x, the proposed sample x' is accepted with probability

$$\alpha(x \to x') = min\left\{1, \frac{q(x|x')\ p(x')}{q(x'|x)\ p(x)}\right\} \tag{2.25}$$

where $\alpha(x \to x')$ is the acceptance probability. If the sample x' is not accepted, the state does not change from the previous one x. Intuitively, a proposed sample is more likely to be selected for high values of target density, i.e. $p(x')$, and for cases that are likely to get back to the old state, i.e. $q(x|x')$. The Metropolis-Hastings algorithm is summarized in Algorithm 1.

Algorithm 1: Metropolis-Hastings

Initialize $x^{(1)}$
for $i = 2, \cdots, N$ **do**
    sample $x^{(i)} \sim q(x^{(i)}|x^{(i-1)})$
    compute $\alpha(x^{(i)}|x^{(i-1)}) = min\left\{1, \frac{q(x^{(i)}|x^{(i-1)})\ p(x^{(i-1)})}{q(x^{(i-1)}|x^{(i)})\ p(x^{(i)})}\right\}$
    sample $u^{(i)} \sim \mathcal{U}(0,1)$
    if $u_i < \alpha(x^{(i)}|x^{(i-1)})$ **then**
        accept $x^{(i)}$
    else
        reject: $x^{(i)} = x^{(i-1)}$

Detailed Balance: This procedure implicitly defines the following transition kernel K:

$$K(x|x') = q(x|x')\ \alpha(x|x') + \delta(x - x')\ \rho(x') \tag{2.26}$$

$$\rho(x') = \int (1 - \alpha(x|x'))\ q(x|x')\ dx \tag{2.27}$$

where $\delta(\cdot)$ is the Dirac delta and $\rho(x')$ is the rejection probability of x' by summing all the probability of rejection. In practice, we do not evaluate K, but this is useful for showing the stationarity of $p(x)$ through detailed balance property. A stationary distribution should satisfy the following condition:

$$p(x) = \int K(x|x')\ p(x')\ dx' \tag{2.28}$$

For an arbitrary kernel K, it is hard to verify stationarity. If K satisfies the following detailed balance condition

$$K(x'|x)\ p(x) = K(x|x')\ p(x') \tag{2.29}$$

then $p(x)$ is a stationary distribution. Integrating both sides over x', we obtain the stationarity condition as defined in Eq. 2.28. Verification of the detailed balance property for the Metropolis-Hastings kernel is straightforward starting from the Eq. 2.26.

For a given target distribution $p(x)$, assuming a proposal distribution $q(x)$, we can use the algorithm in Algorithm 1 to sample from Markov Chain with transition kernel K whose stationary distribution is the target distribution $p(x)$.

Gibbs Sampling

Metropolis-Hastings provides a general framework for deriving MCMC algorithms. The Gibbs sampler is a special case which is developed for sampling a multivariate random variable $\mathbf{x} = (x_1, \cdots, x_D)$ with intractable joint target density $p(x_1, \cdots, x_D)$. The idea is to partition the set of variables into a set of chosen variables $\mathbf{x}_d$ and the rest $\mathbf{x}_{-d}$. While the joint density is intractable, full conditional densities $p(\mathbf{x}_d|\mathbf{x}_{-d})$ are assumed to be tractable. Then, we can use the algorithm in Algorithm 2 by sampling from full conditionals.

Algorithm 2: Gibbs Sampler

```
Initialize x^(1)
for i = 2, ..., N do
    for k = 1, ..., D do
        sample x_k^(i) ~ p(x_k^(i) | x_1^(i), ..., x_{k-1}^(i), x_{k+1}^(i-1), ..., x_D^(i-1))
```

Considering one variable at a time, the Gibbs sampler corresponds to Metropolis-Hastings with a sequence of proposals for $d = 1, \cdots, D$:

$$q_d(\mathbf{x}|\mathbf{x}') = p(x_d|\mathbf{x}'_{-d})\ \delta(\mathbf{x}_{-d} - \mathbf{x}'_{-d}) \tag{2.30}$$

each with a Gibbs transition kernel K_d. Using this proposal distribution, it can be shown that all Gibbs moves are accepted. Each kernel produces the same distribution as the stationary distribution. While the individual transition kernels are not irreducible, after sweeping over all the variables, the resulting effective kernel is aperiodic and irreducible unless the full conditionals are not degenerate. Variables can be visited in deterministic or random order.

Chapter 3

Related Work

This section describes the most relevant related work with respect to feature learning and the use of semantics in reconstruction. We first provide an overview of related feature learning approaches for correspondence estimation with a particular focus on stereo and optical flow estimation (Section 3.1). In the second part, we review approaches which include semantics in the reconstruction either from appearance cues implicitly or from explicit object knowledge (Section 3.2). We finally discuss a number of related approaches from other domains which use higher order potentials or perform joint alignment and appearance estimation (Section 3.3).

3.1 Feature Learning for Correspondence Estimation

Motivated by the success of deep learning in image classification and object recognition (Krizhevsky *et al.*, 2012), a number of papers have tackled the problem of correspondence estimation by learning deep convolutional representations. These approaches learn per-pixel feature representations, typically using Siamese networks which can be fed into a winner-takes-all selection scheme or -as in our case- into a discrete optimization algorithm. While these per-pixel representations tend to be more local compared to end-to-end approaches (Dosovitskiy *et al.*, 2015), they are also less prone to over-fitting. Importantly, even small datasets such as KITTI (Geiger *et al.*, 2012) or MPI Sintel (Butler *et al.*, 2012) provide millions of training points as every pixel provides a training example. This is in contrast to end-to-end approaches where tens of thousands of images with ground-truth flow maps are required for obtaining reliable representations.

Prior to feature learning approaches, Weinzaepfel *et al.* (2013) proposed a descriptor matching algorithm for optical flow, known as DeepMatching. DeepMatching builds quasi-dense correspondence fields by computing similarities of non-rigid patches, allowing for some deformations. In the end, produced matches are combined with a variational approach, similar to Brox and Malik (2011). In contrast to deep learning approaches, DeepMatching does not learn any feature representations, but directly computes correlations at the patch level with fixed convolutions followed by max-pooling. In EpicFlow (Revaud *et al.*, 2015), matches are computed using DeepMatching.

Siamese networks consist of two sub-networks with shared weights and a final score

computation layer. The idea is to train the network for computing the matching cost by learning a similarity measure on small image patches. Žbontar and LeCun (2016) define positive and negative examples as matching and non-matching patches and use a margin loss to train either a fast architecture with a simple dot-product layer in the end or a slow but more accurate architecture which learns score computation with a set of fully connected layers. Luo *et al.* (2016a) use a similar architecture, but formulate the problem as multi-class classification over all possible disparities to capture correlations between different disparities implicitly. Chen *et al.* (2015b) highlight the importance of patch size and propose a multi-scale embedding model by using two different scales with scores merged by a convolutional layer for a weighted ensemble. The multi-scale approach performs better than other baselines using a single scale according to the WTA result.

For the problem of binocular stereo matching, Žbontar and LeCun (2016); Chen *et al.* (2015b); Luo *et al.* (2016a) have demonstrated state-of-the-art performance by combining deep feature representations with discrete optimization. In similar spirit, Zagoruyko and Komodakis (2015) learn Siamese matching networks with an L_2 loss which aims for a broader set of appearance changes in applications such as wide baseline stereo, feature matching and image retrieval. In addition to a regular Siamese network, a variety of different neural network models are explored, namely pseudo-Siamese without sharing weights and a 2-channel network without branches by using concatenated images as input. Furthermore, they investigate the effect of multi-resolution processing with a two-stream network where one is focused on the center of the image. In their experiments, 2-channel-based models show better performance, however, their application to stereo or flow requires increased number of evaluations.

A number of approaches (Han *et al.*, 2015; Simo-Serra *et al.*, 2015) aim for descriptor learning for sparse feature matching. Due to the relatively small number of interest points per image, metric learning networks with fully connected layers can be exploited for this task. Furthermore, Simo-Serra *et al.* (2015) explore a mining strategy for both positive and negative samples to improve the discriminative capability of learned descriptors. Large mining factors, i.e. the ratio of hard examples selected by increasing the batch size, result in improved performance but also incur a high computational cost.

In Gadot and Wolf (2016), Siamese networks for optical flow computation have been combined with winner-takes-all matching and smoothing of the resulting correspondence field. They use a loss that includes batch statistics and a pixel-wise batch normalization which is computationally expensive during test time. While they use patch-wise max pooling operations to increase the size of the receptive field, we exploit computationally efficient dilated convolutions for this purpose. Furthermore, we investigate the usefulness of spatial smoothness priors which we incorporate into dense correspondence estimation via discrete optimization.

Very recently, Bailer *et al.* (2017) proposed a modified hinge embedding loss with threshold for training Siamese networks for optical flow. Similar to Simo-Serra *et al.* (2015), a hard mining strategy is employed by only adding training samples with a non-

zero loss to the batch. A thresholded loss combined with hard mining increases training speed and outperforms other commonly used losses such as the hinge loss in their evaluations. By using FlowFields (Bailer *et al.*, 2015) as basis for their approach, Bailer *et al.* (2017) obtain state-of-the-art results on benchmarks. Instead of improving loss or training strategy, Xu *et al.* (2017) focus on runtime by adapting semi-global matching (Hirschmüller, 2008) to the four-dimensional setting on top of the cost volume created from learned matches. They obtain better results with a significantly reduced runtime.

3.2 Reconstruction and Recognition

Lately, models integrating appearance information into the reconstruction problem have gained popularity. The motivation for incorporating semantics into the reconstruction is that photo-consistency fails in cases of imperfect and ambiguous image information due to specularities, lack of texture, repetitive structures, or strong lighting changes. Some of these cues are based on prior knowledge. For example, semantic labels provide geometric cues about likely surface orientations at a certain location or about the shape of objects commonly observed in the scene. They are often referred as top-down cues, since many of them rely on contextual information which cannot be inferred from patches. There are exceptions which obtain shape information on the patch-level using a data-driven search (Wei *et al.*, 2014; Owens *et al.*, 2013) or clustering (Fouhey *et al.*, 2013). In most cases, these cues contain complementary information which helps resolving matching ambiguities.

3.2.1 Multi-view Stereo

Häne *et al.* (2013) leverage the fact that semantics and surface orientation are mutually dependent, e.g. the surface normal of the ground is more likely to face upwards than downwards. Volumetric scene reconstruction typically segments the volume into occupied and free-space regions. Häne *et al.* (2013) present the mathematical framework and extend it to a multi-label volumetric segmentation framework which assigns object classes or a free-space label to voxels. They first learn appearance likelihoods and class-specific geometry priors for surface orientations from the training data. Then, these data-driven priors are used to define unary and pairwise potentials in a continuous formulation for volumetric segmentation. Joint reasoning benefits from typical class-specific geometry. In addition, it provides a class-specific smoothness prior in cases of weak cues for the scene geometry. Their evaluation shows the benefit of such a prior over standard smoothness assumptions such as Total Variation (Zach, 2008). This knowledge helps in particular for object classes with a dominant orientation (e.g., ground) but has less advantages for classes with a more uniform distribution of normals (e.g., building, car).

Kundu *et al.* (2014) directly constrain the range of possible depth values by conditioning ray potentials on the semantic class. Using a monocular image stream as input, they propose another joint reasoning approach over a sparse point cloud from SfM and dense semantic labeling of the frames. This way, the 3D semantic representation is temporally coherent without additional cost. They model the problem with a higher order CRF in 3D which allows realistic scene constraints and priors such as 3D object support. In addition, they explicitly model the free space which provides cues to reduce ambiguities, especially along weakly supported surfaces. Their evaluation shows improved 3D structure compared to traditional SfM and state-of-the-art multi-view stereo as well as better segmentation quality over video segmentation methods in terms of both per pixel accuracy and the temporal consistency.

Large-scale: Previous works on semantic reconstruction (Häne *et al.*, 2013; Kundu *et al.*, 2014) are limited to small scenes and low resolution, because of their large memory footprint and computational cost. Vineet *et al.* (2015) emphasize the importance of incremental, real-time processing for robotics settings which require large-scale outdoor mapping and propose a semantic reconstruction approach using the hash-based fusion approach of Nießner *et al.* (2013) and volumetric mean-field inference. Blaha *et al.* (2016) point out that high resolution is not required for large regions such as free space, parts under the ground or inside the building. Based on that observation, they propose an extension of Häne *et al.* (2013) by employing an adaptive octree data structure with coarse-to-fine optimization. Starting from a coarse voxel grid, they solve a sequence of problems in which the solution is gradually refined only near the predicted surfaces. The adaptive refinement saves memory and runs much faster while still being as accurate as the fixed voxel discretization at the highest target resolution, both in geometric reconstruction and semantic labeling.

Besides the spatial extent, the number of different semantic labels is also a problem for scalability due to increasing memory requirements. The complexity is quadratic in the number of labels due to indicator variables for the transitions between the different labels. Cherabier *et al.* (2016) propose to divide the scene into blocks in which only a set of relevant labels is active, since absence of many semantic classes from a specific block can be determined early on. Accordingly, they can deactivate a label right from the beginning of the optimization which leads to more efficient processing. The set of active labels in each block is updated during the iterative optimization to recover from wrong initializations. Their evaluation shows that they can increase the number of labels from six to nine with a significant gain in memory compared to Häne *et al.* (2013).

3.2.2 Binocular Stereo

While the mutual benefits of recognition and reconstruction have been shown using simple priors and multiple views (Häne *et al.*, 2013; Kundu *et al.*, 2014), little work has

addressed the binocular stereo problem in this context with notable exceptions. Saxena *et al.* (2007) propose to combine monocular cues such as texture variations, texture gradients and color with binocular cues in a multi-scale MRF model. These depth-from-appearance constraints are directly integrated into the data term. They conclude that monocular cues and geometric stereo cues contain complementary information about depth and using both improves accuracy significantly compared to either monocular or stereo cues alone. In contrast, Ladicky *et al.* (2010) model stereo estimation and semantic segmentation jointly by learning the dependency between the height over ground and the semantic class. Their evaluation shows that joint optimization of the two problems improves performance of stereo estimation by resolving ambiguities.

Data-Driven: Wei *et al.* (2014) propose a data-driven regularization to transfer the shape information of the disparity or flow from semantically matched patches in the training database using the SIFT flow algorithm of Liu *et al.* (2008). They represent the shape information as the relative relationship of scene properties instead of absolute values for reusability of scene properties, such as modeling the disparity of a car independent of its position. The data-driven prior shows improved performance against popular smoothness terms on benchmarks. Unfortunately, the nature of interaction in these models is very local and thus cannot constrain large ambiguous regions well enough. Karsch *et al.* (2014) propose a depth transfer approach on the videos using temporal information from motion estimation. Using SIFT flow, a set of warping functions is estimated from semantically similar candidate images with known depth maps. Depth estimation is performed as an energy minimization considering all of the warped candidates. In general, depth transfer approaches require a representative training set with available depth maps due to the critical assumption that the distribution of depth is comparable among the input and the training set.

3.2.3 Single Image Depth Estimation

In the extreme case, depth estimation can be performed without any binocular cues using a single image as input. One of the earliest approaches, Hoiem *et al.* (2005) estimate the coarse geometric properties of a scene by learning appearance-based models of coarse geometric classes such as ground, sky and vertical regions. Saxena *et al.* (2009) propose a segment-based approach known as Make3D that first over-segments the input image into approximately planar surfaces and then estimates the 3D location and orientation of each using a MRF. Fouhey *et al.* (2013) propose to infer 3D surface normals given a single image. Their approach is based on clustering image patches where each cluster is discriminative and geometrically consistent with similar surface normals. Learned cluster parameters are used as detectors during test time and the surface normal of a pixel is computed as a linear combination of the detected surface normals. Ladicky *et al.* (2014b) propose a semantic depth classifier to predict a fixed canonical depth conditioned

on the semantic label. Such a classifier makes use of semantics as well as the perspective geometry, i.e. the perceived size of the objects scales inversely with the distance.

Deep Learning: Single image depth estimation approaches have shown great progress with the use of deep learning. Liu *et al.* (2015) propose a unified deep CNN framework to jointly learn the unary and pairwise potentials of a continuous CRF for single depth estimation. Wang *et al.* (2015) extend the surface estimation approach proposed in Fouhey *et al.* (2013) by combining a bottom-up network that maps local patches to their local orientation and a top-down network that takes the whole image as input for global cues. Without relying on an initial segmentation, Eigen and Fergus (2015); Eigen *et al.* (2014) propose to use a two-scale network to predict depth, surface normals and semantic labels for each pixel from a single image. Inspired by the success of this method, there are follow-up works using CRFs to improve accuracy (Li and Chen, 2015), changing the loss from regression to classification (Cao *et al.*, 2016) and using other more robust loss functions (Laina *et al.*, 2016). In contrast to supervised methods which require vast amounts of ground-truth data, Godard *et al.* (2017) propose an unsupervised monocular depth estimation approach using a loss function based on left-right consistency check.

3.2.4 Object Knowledge

Appearance cues can be incorporated into the reconstruction by explicitly recognizing certain objects and modeling their shape. Object information can be useful to enforce some simple physical constraints based on prior knowledge or to ignore object regions in certain cases such as simplified geometries which are often violated by some objects. Furthermore, the shape of objects can be explicitly modeled using linear or non-linear shape embeddings, parametric models or external 3D models. These models can be used as shape priors to improve 3D reconstruction or they can be learned while reconstructing the scene.

Binocular Stereo

Bleyer *et al.* (2011) extend the low-level segmentation of Bleyer *et al.* (2010) to the object level by enforcing 3D connectivity of objects to better handle difficult occlusion cases. While Bleyer *et al.* (2011) assume that objects are approximately planar, Bleyer *et al.* (2012) use enclosing 3D bounding boxes to introduce and exploit physical constraints such as occupancy, i.e. non-overlapping boxes and gravity. Unaware of the object class, a large pool of object proposals are generated and ranked according to a learned objectness score. Then, object labeling and a disparity map are jointly inferred on the top ranked object proposals. While promising, such regularizers ignore the semantic context which heavily constrains the shape of the geometric primitives.

Shape Priors

In case of simplified assumptions on the geometry, e.g. ruled surfaces, objects such as cars which are prevalent in urban scenes violate these assumptions. Cornelis *et al.* (2008) integrate the detection and localization of cars into the reconstruction to handle these cases. While they just instantiate virtual placeholder models on the detections, a more general approach is to model the shape of objects as geometric priors. Dimensionality reduction is an effective and popular way of representing the shape knowledge. Early approaches use linear dimensionality reduction such as PCA to capture the shape variance in low dimensional latent shape spaces. More recent approaches use nonlinear dimensionality reduction such as Gaussian Process Latent Variable Models (GP-LVM). Prisacariu and Reid (2011a,b) demonstrate impressive results by leveraging GP-LVMs for learning a non-linear TSDF embedding of the object shape for segmentation and reconstruction.

Object shape priors have been used to improve dense 3D reconstruction. Dame *et al.* (2013) investigate the importance of shape priors in a monocular SLAM approach. In parallel with depth estimation, they refine an object's pose, shape and scale to match an initial segmentation and depth cues. It is finally fused into the volumetric representation. Their experiments show improvement in transparent and specular surfaces and even in unobserved parts of the scene. In addition to the mean shape, Bao *et al.* (2013) propose to learn a set of anchor points as representative of object shape across several instances. They first perform an initial alignment using 2D object detectors. Next, they align the point cloud from SfM with the mean shape by matching anchor points, and then warp and refine it to approach the actual shape. Their evaluation demonstrates that the model is general enough to learn semantic priors for different object categories such as car, fruit and keyboard by handling large shape variations across instances. While Dame *et al.* (2013) and Bao *et al.* (2013) try to fit a parametric shape model to input data, Häne *et al.* (2014a) model the local distribution of normals for an object. They propose an object class specific shape prior in the form of spatially varying anisotropic smoothness terms.

Similar techniques have been successfully applied for updating the shape model during object tracking. The pixel-wise posterior tracker (PWP) by Bibby and Reid (2008) which is a probabilistic 2D multiple-object tracking and segmentation method, has been extended to include shape and semantic priors (Prisacariu and Reid, 2012; Ma and Sibley, 2014). Salas-Moreno *et al.* (2013) present a SLAM system incorporating 3D object detection for indoor mapping. The poses of the detected 3D objects, along with the poses of the camera, are jointly optimized in a graph optimization framework. While these methods require known a priori 3D object models, Ren *et al.* (2013); Ma and Sibley (2014) simultaneously track an object while reconstructing a 3D model. In case of Ma and Sibley (2014), multiple objects can be simultaneously tracked and reconstructed.

3.3 Other Domains

Our method based on object disparity hypotheses for stereo borrows ideas from binary segmentation approaches leveraging pattern-based potentials (Rother *et al.*, 2009; Komodakis and Paragios, 2009; Marquez-Neila *et al.*, 2014) to encourage plausible label configurations. While our object disparity proposals can be interpreted as pattern potentials, we differ in that we optimize a continuous label space and model interactions far beyond the typical 10×10 pixel patches used in segmentation approaches.

In contrast to models assuming piecewise planarity or a Manhattan world, our approach also applies to non-planar object classes such as cars. In this sense, our ideas are related to the depth super-resolution method of Hornacek *et al.* (2013). However, while they tackle single depth images using a patched-base representation, we model complete 3D scenes at the object level.

Related to our joint reconstruction approach are also a number of techniques which consider joint image alignment and segmentation or appearance estimation (Cremers *et al.*, 2007) with applications in face recognition (Deng *et al.*, 2014) and medical imaging (Tsai *et al.*, 2005). Our approach shares similarity with these methods in terms of estimating an instance specific transformation jointly with a low-dimensional representation of the object. However, our focus is on 3D reconstruction rather than 2D segmentation and we do not assume that objects are presented in a stereotypical pose. Instead, our method localizes objects according to a discriminatively trained object proposal generator. Furthermore, our model handles missing information (e.g., unobserved voxels) and is able to deal with a broad range of shapes by clustering them into different model components.

Chapter 4

Stereo Estimation using Object Knowledge

In this chapter, we investigate the utility of object recognition and semantic segmentation for stereo matching. In particular, we focus our attention on the reconstruction of geometrically well-defined objects, cars, for which the data term is weak and current methods perform poorly. Due to their textureless, reflective and semi-transparent surface, car regions in the image represent a major challenge for stereo algorithms, as illustrated in Fig. 4.1 (top).

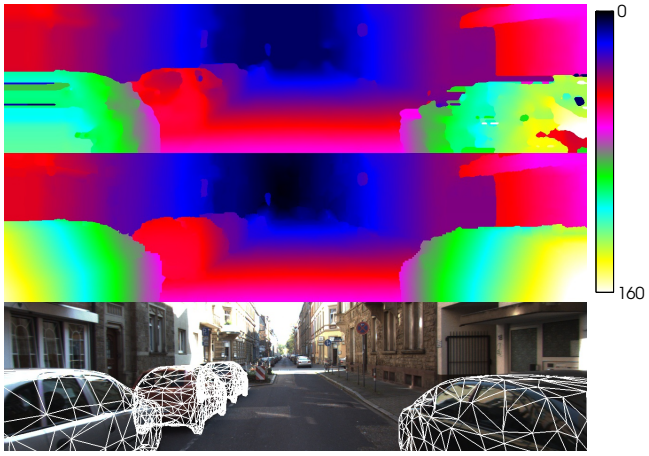


Figure 4.1: **Resolving Stereo Matching Ambiguities:** Stereo methods often fail at reflecting, textureless or semi-transparent surfaces (top, Zbontar and LeCun (2014)). By using object knowledge, we encourage disparities to agree with plausible surfaces (center). This improves results both quantitatively and qualitatively while simultaneously recovering the 3D geometry of the objects in the scene (bottom).

While the reconstruction of purely specular surfaces has been successfully demonstrated using multiple frames Weinmann *et al.* (2013); Jacquet *et al.* (2013), such techniques are hard to employ in our setting due to the superposition of several deteriorating visual effects in real-world scenes.

In contrast, as humans we are able to effortlessly extract information about the geometry of cars even from a single image using our prior object knowledge and shape representation. Inspired by this fact, we introduce object knowledge for well-constrained object categories into a slanted-plane MRF and estimate a dense disparity map. Towards this goal, we leverage semantic information and inverse graphics to sample a set of plausible object disparity maps given an initial semi-dense disparity estimate (Section 4.1). We encourage the presence of these 2.5D shape samples (or "displets") in our MRF formulation depending on how much their geometry and semantic class agrees with the image observations (Section 4.2). Our experiments indicate that the proposed framework is able to resolve stereo ambiguities on challenging stereo pairs from the KITTI benchmark (Section 4.3). In addition, our method is able to extract 3D object representations which are consistent with the estimated disparity map, see Fig. 4.1 for an illustration.

4.1 Object Knowledge

In this section, we explain how we obtain the prior object knowledge as a first step to the joint inference. We form our prior on the location and geometry of the object category to restrict the possible locations and shape variations of the object in a particular scene. Ideally, the prior should be representative of the object category and should agree with the image evidence, both semantically and geometrically. Furthermore, the solution should be able to handle multiple instances of the object.

Given a semi-dense initial disparity map and a rough segmentation of the object on the image, we generate a set of object disparity proposals, i.e. *displets*, using rapid inverse graphics techniques. Intuitively, displets can be considered as a representative finite subset of the infinitely large set of disparity maps for that class conditioned on the image. For example, car displets should cover the most likely 3D car shapes given the two input images.

We start with a set of representative CAD models from Google 3D Warehouse[1] which capture most of the 3D shape variability of the object category as shown in Fig. 4.2. In this case, a small set of 3D CAD models suffices due to the restrictive geometry of the object class car, see Section 4.3 for an experiment on varying the number of CAD models used. In Section 4.1.1, we propose a simple method which reduces the details of a 3D model while preserving the hull of the object in order to speed up the rendering process. In Section 4.1.2, we define an observation model to subsample the space of plausible displets using inverse graphics and provide details of the sampling procedure.

4.1.1 Simplification: Semi-Convex Hull

Unfortunately, CAD models downloaded from Google Warehouse are not directly applicable as they are often designed with love of detail resulting in hundreds of thousands

[1] `https://3dwarehouse.sketchup.com/`

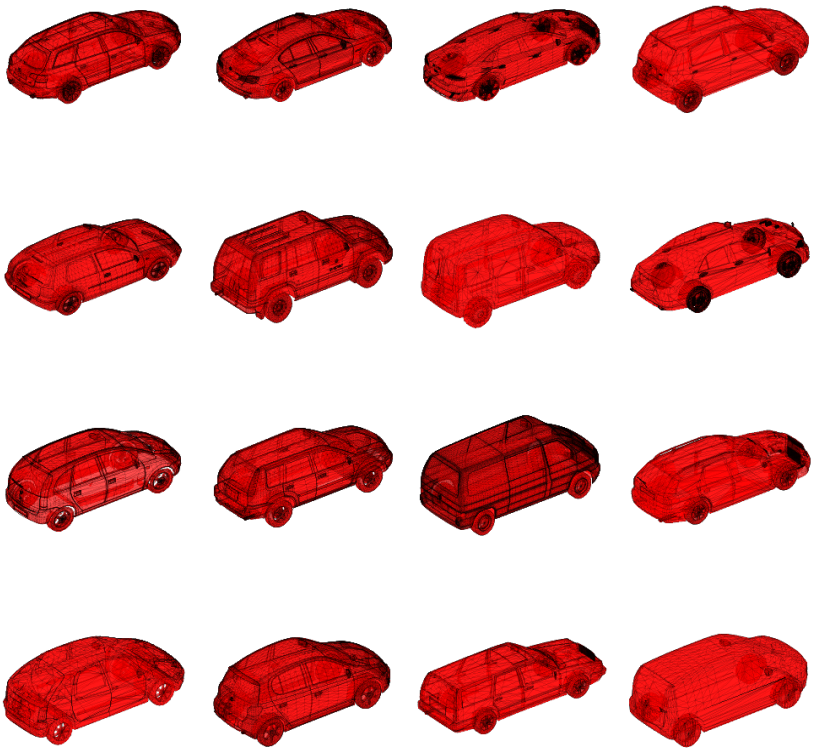


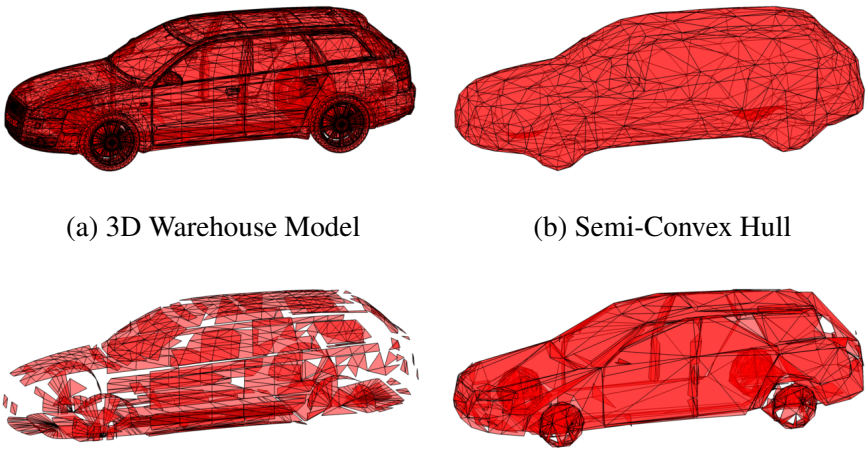
Figure 4.2: **3D CAD Models.** This figure shows the set of representative CAD models from Google 3D Warehouse which capture most of the 3D shape variability for the object car.

of vertices and faces, slowing down the rendering process significantly. We found that tools like MATLAB's `reducepatch` function[2] or QSlim (Garland and Heckbert, 1997) are not able to simplify these models to an affordable level of detail without introducing holes or artifacts as illustrated in Fig. 4.3.

In this section, we propose a simple method which reduces a CAD model of geometrically simple classes such as cars to around 1000 faces while preserving the hull of the object and removing all interior elements which do not affect the rendered depth map. We initialize a mesh using the convex hull of the object and gradually relax it to a smooth approximation, subject to the constraint that the volume of the model comprises all surface points. We call this representation the "semi-convex hull" of an object. In particular, we minimize the squared point-to-point distances between all vertices of the mesh and densely sampled points on the original 3D model.

Let $\mathcal{P}$ denote the set of 3D points obtained by uniformly sampling a large number of

[2] `http://www.mathworks.de/help/matlab/ref/reducepatch.html`



(a) 3D Warehouse Model (b) Semi-Convex Hull

(c) QSlim (d) MATLAB

Figure 4.3: **Mesh Simplification.** For efficient rendering, we simplify 3D CAD Models with ∼100K faces (a) by a smooth semi-convex approximation using 1K faces only (b). The application of generic mesh simplification algorithms using the same number of faces produces undesirable holes and self-intersections in the mesh as illustrated in (c+d).

3D points, i.e. 50K, from the union of all surfaces of the object (full resolution CAD model). For uniform sampling of points, we compute the area of each triangle and construct a list of normalized area weights, i.e. the area of a triangle divided by the total surface area. We treat this list as a discrete probability distribution to draw a set of triangles with respect to the surface area. For each triangle with vertices $(\mathbf{a},\mathbf{b},\mathbf{c})$, we sample a point on its surface by the following convex combination of the vertices:

$$\mathbf{p} = (1-\sqrt{r_1})\mathbf{a} + \sqrt{r_1}(1-r_2)\mathbf{b} + \sqrt{r_1}r_2\mathbf{c} \tag{4.1}$$

where r_1 and r_2 are two random numbers between 0 and 1.

Let further $\mathcal{M} = \{\mathcal{V},\mathcal{F}\}$ denote a mesh with vertices $\mathcal{V}$, faces $\mathcal{F}$ and edges $\mathcal{E}(\mathcal{F})$. Each vertex $i \in \mathcal{V}$ is associated with a variable $\mathbf{x}_i \in \mathbb{R}^3$ specifying the location of the vertex. We first obtain the object's convex hull as the exterior face of the Delaunay triangulation. We initialize $\mathcal{M}$ and $\mathbf{x} = \{\mathbf{x}_i | i \in \mathcal{V}\}$ by uniformly remeshing the object's convex hull using isotropic surface remeshing (Fuhrmann *et al.*, 2010) and formulate our objective as minimizing

$$\begin{aligned} E(\mathbf{x}) &= \sum_{i\in\mathcal{V}} \|\mathbf{x}_i - nn(\mathbf{x}_i)\|^2 + \sum_{(i,j)\in\mathcal{E}} \left(\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \bar{l}_0\right)^2 \\ \text{s.t.} \quad & \mathcal{P} \subseteq \text{Vol}(\mathcal{M},\mathbf{x}) \end{aligned} \tag{4.2}$$

where $nn(\mathbf{x}_i)$ denotes the nearest neighbor of $\mathbf{x}_i$ in $\mathcal{P}$, $\bar{l}_0$ is the average edge length of the initial mesh, and $\text{Vol}(\mathcal{M},\mathbf{x})$ denotes the set of 3D points inside the mesh. We solve Eq. 4.2 using gradient descent and enforce the closure constraint $\mathcal{P} \subseteq \text{Vol}(\mathcal{M},\mathbf{x})$ by mesh rescaling and uniform remeshing on violation. After convergence, we obtain a smooth semi-convex hull of the object which we simplify to 1K faces using QSlim (Garland and Heckbert, 1997). See Algorithm 3 for further details.

Algorithm 3: Mesh Simplification

Input: 3D CAD model
Output: Semi-convex hull $(\mathcal{M},\mathbf{x})$

$\mathcal{P} \leftarrow$ draw samples from 3D CAD model
$(\mathcal{M},\mathbf{x}) \leftarrow$ convex hull of 3D CAD model
$(\mathcal{M},\mathbf{x}) \leftarrow$ remeshing of $(\mathcal{M},\mathbf{x})$ using Fuhrmann *et al.* (2010)
while *not converged* **do**
  $\mathbf{x} \leftarrow \mathbf{x} - \gamma \nabla E(\mathbf{x})$
  if $\mathcal{P} \nsubseteq Vol(\mathcal{M},\mathbf{x})$ **then**
    $\alpha \leftarrow \min(\{\alpha > 0 \,|\, \mathcal{P} \subseteq \text{Vol}(\mathcal{M}, \alpha\mathbf{x})\})$
    $\mathbf{x} \leftarrow (\alpha + \varepsilon)\mathbf{x}$
    $(\mathcal{M},\mathbf{x}) \leftarrow$ remeshing of $(\mathcal{M},\mathbf{x})$ using Fuhrmann *et al.* (2010)
$(\mathcal{M},\mathbf{x}) \leftarrow$ simplification of $(\mathcal{M},\mathbf{x})$ using Garland and Heckbert (1997)

4.1.2 Sampling Displets

This section describes how we subsample the infinitely large space of disparity maps using inverse graphics, yielding the set of displets $\mathcal{D}$ which will be used in the inference as explained in the next section. Vision as inverse graphics is based on the idea of inverting the image generation process by modeling the factors that produce different scene configurations. A classical application of inverse graphics is inferring 3D shapes from single images. In that case, generative models need to account for wide variability in 3D shape and appearance in 2D. Therefore, the inversion problem is computationally intractable and sampling techniques are often used for approximate inference. Similarly, we make use of MCMC to draw a set of representative samples corresponding to a certain object category (e.g., cars).

In a typical inverse graphics approach, the image returned by the renderer is compared to the observed image, for example via a pixel-wise Gaussian likelihood model (Mansinghka *et al.*, 2013). In contrast, our generative process produces disparity maps from CAD models using the camera intrinsics. Our likelihood model compares the rendered disparity map with the input disparity map $\hat{\Omega}$ and returns a score depending on the level of agreement. This makes our algorithm invariant to the actual image intensities

which are hard to model in a generative way, in particular in the presence of reflecting or translucent surfaces.

For a given object category c, we are interested in sub-sampling the space of plausible displets given a semi-dense disparity image $\hat{\Omega}$, a semantic segmentation $\mathbf{S}$, and the semi-convex hull of all CAD models of this object category. We approach this inverse graphics problem using MCMC, i.e., we sample pose parameters $\xi \in SE(3)$ directly from the observation model $p(\xi|\hat{\Omega}) \propto \exp(-E_{\hat{\Omega}}(\xi))$ with

$$\begin{aligned} E_{\hat{\Omega}}(\xi) &= \sum_{\mathbf{p} \in \hat{\Omega}_+ \cap \mathcal{O}} \frac{\min\left(|\bar{\omega}(\mathbf{p},\xi) - \hat{\omega}(\mathbf{p})|, \tau_1\right)}{|\hat{\Omega}_+ \cap \mathcal{O}|} \\ &\quad + \beta \sum_{\mathbf{p} \in \hat{\Omega}_+} [\bar{\omega}(\mathbf{p},\xi) > \hat{\omega}(\mathbf{p}) + \tau_2] \end{aligned} \tag{4.3}$$

Here, $\mathcal{O}$ denotes a 2D object instance in the image, $\bar{\omega}(\mathbf{p},\xi)$ is the disparity of the CAD model in pose ξ rendered at pixel $\mathbf{p}$, and $\beta, \tau_1, \tau_2 > 0$ are parameters of the model. Intuitively, $E_{\hat{\Omega}}(\xi)$ encourages displets to explain all pixels within object region $\mathcal{O}$ in terms of disparity (first term) while avoiding the occlusion of other objects (second term).

In principle, the use of object proposals $\mathcal{O}$ could be avoided by directly sampling according to the semantic labeling $\mathbf{S}$, but we found instance level information to improve the diversity of the displet set. While a large number of generic object proposal algorithms (Krähenbühl and Koltun, 2014; Zitnick and Dollár, 2014; Carreira and Sminchisescu, 2012; Uijlings *et al.*, 2013) can be applied to obtain the set of object proposal regions $\{\mathcal{O}\}$, we found a much more simple strategy to be sufficient for our goals: First, we project all valid pixels of class c ($\mathbf{p} \in \hat{\Omega}_+ \cap [\mathbf{S} = c]$) into 3D. Next, we apply kernel density estimation (KDE) along the principal coordinate axes x and z of the camera. As object boundaries frequently coincide with minima of the KDE, we propose one object region $\mathcal{O}$ for each pair of adjacent minima by projecting all 3D points in this range back into the image. It is important to note that we do not assume precise object boundaries for the proposals due to the robust term in Eq. 4.3.

We run one Markov chain for each combination of CAD models and object proposals using Metropolis-within-Gibbs (MWG) sampling (5.000 iterations) with randomly chosen blocks. For each combination, we select the 8 most dominant modes after burn-in and combine all results to yield the final set of displets. As our semi-convex mesh has a low number of vertices and faces, we are able to draw a large number of samples on commodity graphics hardware. In practice, we achieve $\sim$ 8200 fps on a single NVIDIA Quadro 4000 GPU using 12 threads. Fig. 4.4 visualizes a subset of the sampled displets for a single test image which serve as input to our CRF. Wrong displet proposals are eliminated during inference if they do not agree with the observations as explained next.

Figure 4.4: **Illustration of Displets.** A subset of sampled displets is shown for an image by overlaying the corresponding disparity maps on top of the image regions they have been proposed for.

4.2 Stereo Matching using Displets

We tackle the classical task of binocular stereo matching. That is, given the left and right images of a synchronized, calibrated and rectified stereo camera, we are interested in estimating the disparity at each pixel of the reference image (e.g., left image).

We follow a slanted plane MRF approach and assume that the image can be decomposed into a set of planar superpixels (Section 4.2.1). Superpixel segmentation algorithms typically cluster pixels in the combined five-dimensional color and location to generate segments that are compact and roughly equally sized (Achanta *et al.*, 2012). Given an initial estimate of the depth, superpixels can be further improved to preserve depth discontinuities. The StereoSLIC algorithm of Yamaguchi *et al.* (2013) formulates the segmentation energy based on both image and depth terms which results in an over-segmentation of the image that respects both depth and image boundaries.

We use the StereoSLIC algorithm to obtain superpixels representing the slanted planes in the scene. The number of superpixels is a parameter which we experiment with, see

Section 4.3 for further details and the visualization of superpixels. We define our conditional random field in terms of all slanted-planes and displet variables (Section 4.2) and encode potentials over sets of continuous and discrete variables. In addition to unary (Section 4.2.2) and local pairwise constraints (Section 4.2.3), we also introduce long-range interactions into our model (Section 4.2.4) using *displets*, a set of physically plausible disparity maps of a certain semantic class, as explained in Section 4.1. We use max-product particle belief propagation to perform inference in the resulting mixed continuous-discrete random field (Section 4.2.5).

4.2.1 Representation

In our formulation, each superpixel is represented as a random variable $\mathbf{n}_i \in \mathbb{R}^3$ describing a plane in 3D ($\mathbf{n}_i^T \mathbf{x} = 1$ for $\mathbf{x} \in \mathbb{R}^3$ on the plane). We denote the disparity of plane $\mathbf{n}_i$ at pixel $\mathbf{p} = (u,v)^T$ by $\omega(\mathbf{n}_i, \mathbf{p})$. In the following, we show how $\omega(\mathbf{n}_i, \mathbf{p})$ can be derived for a (rectified) pinhole stereo camera with known intrinsics and extrinsics:

$$\begin{aligned} u &= \frac{fx}{z} + c_u, \;\; x = (u - c_u)\frac{z}{f} \\ v &= \frac{fy}{z} + c_v, \;\; y = (v - c_v)\frac{z}{f} \\ d &= \frac{fL}{z} \end{aligned} \qquad \begin{aligned} 1 &= x\mathbf{n}_x + y\mathbf{n}_y + z\mathbf{n}_z \\ 1 &= (u - c_u)\frac{z}{f}\mathbf{n}_x + (v - c_v)\frac{z}{f}\mathbf{n}_y + z\mathbf{n}_z \\ \frac{1}{z} &= (u - c_u)\frac{\mathbf{n}_x}{f} + (v - c_v)\frac{\mathbf{n}_y}{f} + \mathbf{n}_z \end{aligned}$$

$$\begin{aligned} d &= (u - c_u)L\mathbf{n}_x + (v - c_v)L\mathbf{n}_y + fL\mathbf{n}_z \\ d &= L\mathbf{n}_x u + L\mathbf{n}_y v + fL\mathbf{n}_z - c_u L\mathbf{n}_x - c_v L\mathbf{n}_y \qquad (4.4) \\ d &= au + bv + c \qquad (4.5) \end{aligned}$$

where f, c_u, c_v denote the intrinsic camera calibration parameters and L is the baseline. For a 3D point $\mathbf{x} = (x,y,z)^T$ on plane $\mathbf{n}$, we compute disparity d of plane $\mathbf{n}$ at pixel $\mathbf{p}$, which corresponds to the plane equation in Eq. 4.5. Next, we introduce the notation including the random variables in the energy function.

- Let $\mathcal{S}$ and $\mathcal{D}$ denote the set of superpixels and displets in the reference image.
- Each superpixel $i \in \mathcal{S}$ is associated with
  - a region $\mathcal{R}_i$ in the image,
  - a random variable $\mathbf{n}_i \in \mathbb{R}^3$ describing a plane in 3D.
- Each displet $k \in \mathcal{D}$ is associated with
  - its class label $c_k \in \mathcal{L} \backslash \{\text{background}\}$,
  - a fitness value $\kappa_k \in \mathbb{R}$,

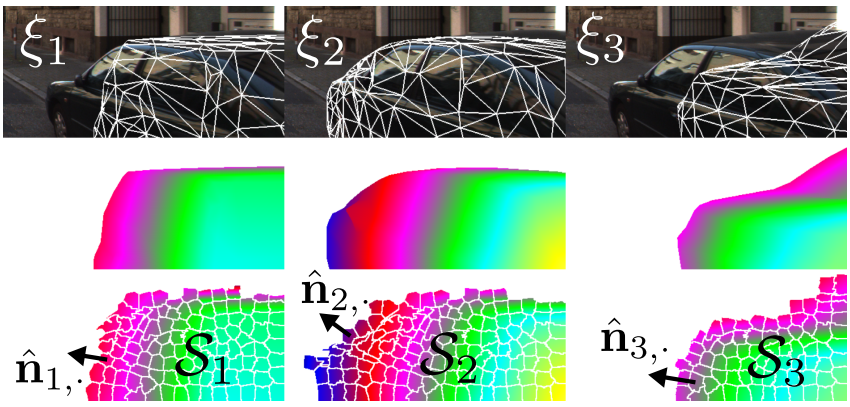


Figure 4.5: **Illustration of Displets:** We sample 3D CAD model configurations ξ_k (top+center) and extract the plane parameters $\hat{\mathcal{N}}_k = (\hat{\mathbf{n}}_{k,1}, \ldots, \hat{\mathbf{n}}_{k,|\mathcal{S}_k|})^T$ of the corresponding displet $k \in \mathcal{D}$ by fitting planes to the rendered disparity map for all involved superpixels $\mathcal{S}_k$ (bottom).

  - a set of superpixels $\mathcal{S}_k \subseteq \mathcal{S}$ on which it is defined,
  - the corresponding plane parameters $\hat{\mathcal{N}}_k = (\hat{\mathbf{n}}_{k,1}, \ldots, \hat{\mathbf{n}}_{k,|\mathcal{S}_k|})^T$.

  The plane parameters are obtained by local plane fitting to the rendered disparity map of the corresponding CAD model (Fig. 4.5).

- An additional random variable $d_k \in \{0,1\}$, which can be interpreted as auxiliary variable in a high-order CRF, denotes the presence ($d_k = 1$) or absence ($d_k = 0$) of the displet in the scene.

- Furthermore, we assume that we have access to a rough semantic segmentation of the image $\mathbf{S} \in \mathcal{L}^{W \times H}$ with $|\mathcal{L}|$ the number of semantic labels[3] and $W \times H$ the image dimensions. We obtain this segmentation using the associative hierarchical CRF method proposed in Ladicky *et al.* (2014a) and refer the reader to Section 4.3 for further details.

Given the left and right image, our goal is to infer all superpixel plane parameters $\mathbf{n}_i$ as well as the presence or absence of all displets d_k in the scene. We specify our CRF in terms of the following energy function

$$\begin{aligned} E(\mathbf{n},\mathbf{d}) \quad = \quad & \sum_{i \in \mathcal{S}} \varphi_i^{\mathcal{S}}(\mathbf{n}_i) + \sum_{i \sim j} \psi_{ij}^{\mathcal{S}}(\mathbf{n}_i, \mathbf{n}_j) + \\ & \sum_{k \in \mathcal{D}} \varphi_k^{\mathcal{D}}(d_k) + \sum_{k \in \mathcal{D}} \sum_{i \in \mathcal{S}_k} \psi_{ki}^{\mathcal{D}}(d_k, \mathbf{n}_i) \end{aligned} \tag{4.6}$$

[3] While keeping our exposition general, we only consider "car" vs. "background" in our experiments as cars are the most challenging object category in KITTI while still sufficiently restricted in terms of geometry.

where $\mathbf{n} = \{\mathbf{n}_i | i \in \mathcal{S}\}$ and $\mathbf{d} = \{d_k | k \in \mathcal{D}\}$ and $i \sim j$ denotes the set of adjacent superpixels in $\mathcal{S}$. Next, we explain each term in the energy.

4.2.2 Data Term

The data term models the fact that corresponding points in the left and right image should be similar in appearance. While many options are possible, we simply penalize deviations from an initial sparse disparity map $\hat{\Omega}$, calculated with a semi-dense feature matching algorithm:

$$\varphi_i^{\mathcal{S}}(\mathbf{n}_i) = \sum_{\mathbf{p} \in \mathcal{R}_i \cap \hat{\Omega}_+} \rho_{\tau_1}(\omega(\mathbf{n}_i, \mathbf{p}) - \hat{\omega}(\mathbf{p})) \tag{4.7}$$

Here, $\hat{\Omega}_+$ denotes the set of valid pixels in $\hat{\Omega}$, $\omega(\mathbf{n}_i, \mathbf{p})$ is the disparity of plane $\mathbf{n}_i$ at pixel $\mathbf{p}$, and $\hat{\omega}(\mathbf{p})$ represents the value of the reference disparity map $\hat{\Omega}$ at pixel $\mathbf{p}$ inside region $\mathcal{R}_i$ corresponding to the superpixel. To account for outliers, we chose $\rho_\tau(\cdot)$ as the robust l_1 penalty function $\rho_\tau(x) = \min(|x|, \tau)$. In Section 4.3, we further evaluate and compare two state-of-the-art feature matching algorithms which yield the initial sparse disparity map $\hat{\Omega}$.

4.2.3 Local Smoothness

We encourage local smoothness in our formulation by penalizing discontinuities at superpixel boundaries as well as by encouraging similar orientations of adjacent superpixels. In particular, our smoothness term decomposes as

$$\begin{aligned} \psi_{ij}^{\mathcal{S}}(\mathbf{n}_i, \mathbf{n}_j) \;\; = \;\; & \theta_1 \sum_{\mathbf{p} \in \mathcal{B}_{ij}} \rho_{\tau_2}\left(\omega(\mathbf{n}_i, \mathbf{p}) - \omega(\mathbf{n}_j, \mathbf{p})\right) \; + \\ & \theta_2 \;\; \rho_{\tau_3}\left(1 - |\mathbf{n}_i^T \mathbf{n}_j| / (\|\mathbf{n}_i\| \|\mathbf{n}_j\|)\right) \end{aligned} \tag{4.8}$$

where $\mathcal{B}_{ij}$ denotes the set of shared boundary pixels between superpixel i and superpixel j and the other functions are defined as above.

The first term minimizes the difference between the disparities proposed by adjacent superpixels on the boundary pixels and the second term minimizes the cosine distance between the normals of adjacent superpixels. The weights θ_1, θ_2 control the importance of each term with respect to the other terms in Eq. 4.6. Inspired by contrast-sensitive smoothness priors, we down-weight θ_1 and θ_2 if neighboring superpixels i and j are likely to be separated by an occlusion boundary. This likelihood is computed by simply detecting large changes in the gradient of the input disparity map $\hat{\Omega}$.

4.2.4 Displet Potentials

In order to encode long-range interactions, we introduce displet potentials which encourage plausible geometries in regions corresponding to a certain semantic class.

Displet Unary

The unary potential for displet d_k is defined as

$$\varphi_k^{\mathcal{D}}(d_k) = -\theta_3 \, [d_k = 1] \cdot (|[\mathbf{S} = c_k] \cap \mathbf{M}_k| + \kappa_k) \tag{4.9}$$

where $[\cdot]$ denotes the (element-wise) Iverson bracket, $\mathbf{M}_k$ represents the pixel mask corresponding to the set of superpixels $\mathcal{S}_k$, and κ_k is a fitness score assigned to displet k. Intuitively, this potential tries to explain as many regions in the image which have been assigned the semantic class label c_k using displets whose shape corresponds to typical objects in class c_k. Furthermore, we take the fitness score κ_k as the displet log likelihood ($-E_{\hat{\Omega}}$) in order to increase the plausibility of high quality displets. In practice, we subtract the largest fitness score of all displets originating from the same object proposal region $\mathcal{O}$ in order to calibrate for the scores of different regions.

Displets and Superpixels

We define a potential between each displet and all the superpixels it comprises as follows

$$\psi_{ki}^{\mathcal{D}}(d_k, \mathbf{n}_i) = \lambda_{ki} \, [d_k = 1] \cdot \big(1 - \delta(\mathbf{n}_i, \hat{\mathbf{n}}_{k,z_i})\big) \tag{4.10}$$

where z_i denotes the index of the plane corresponding to superpixel i in $\mathcal{N}_k$ and $\delta(\cdot,\cdot) = 1$ if both arguments are equivalent and 0 otherwise. This term encourages the superpixels covered by the displet's mask to take the plane parameters proposed by the displet when it is on, i.e. $d_k = 1$. It is important to note that the hard constraint avoids evidence undercounting, i.e., it ensures that displets do not overlap and explain the same region in the image.

Displet Influence: The influence of a displet on its associated superpixels is determined by a penalty function λ_{ki} to better account for inaccuracies in the displet mask $\mathbf{M}_k$. While many choices are possible, we define λ_{ki} as a weighted sigmoid function of the distance transform of $\mathbf{M}_k$

$$\lambda_{ki} = \lambda_\theta \, \frac{1}{1 + \exp(\lambda_a - \lambda_b \, \mathrm{DT}_{ki})}$$

where DT_{ki} denotes the Euclidean distance transform of superpixel i with respect to the boundary of displet k. The model parameters λ_θ, λ_a and λ_b are learned from training data. In Fig. 4.6, we visualize λ_{ki} for a couple of displets and the associated superpixels.

Less transparency indicates a higher penalty, i.e., we allow more deviation at the displet boundaries. In contrast, λ_{ki} takes a very large value (λ_θ) in the center of the displet, effectively approximating a hard constraint ($\lambda_{ki} = \infty$).

4.2.5 Inference

Minimizing Eq. 4.6 is a non-convex mixed continuous-discrete optimization problem which is NP-hard to solve. We leverage greedy max-product particle belief propagation (MP-PBP) (Trinh and McAllester, 2009; Pacheco *et al.*, 2014) with sequential tree-reweighted message passing (TRW-S) (Kolmogorov, 2006) using 30 particles and 50 iterations to find an approximate solution (Algorithm 4). At every iteration, plane particles are sampled from a normal distribution around the previous MAP solution and using the plane parameters of spatially neighboring superpixels. Both strategies complement each other and we found their combination important for efficiently exploring the search space. To ensure that displets are selected with non-zero probability, we augment the proposal set for a superpixel by the plane parameters of all overlapping displets. We initialize all superpixel planes using the StereoSLIC algorithm (Yamaguchi *et al.*, 2013).

Algorithm 4: MP-PBP for Stereo Estimation using Displets

Initialize planes $\mathbf{n}_i^{(0)} = (a_i^{(0)}, b_i^{(0)}, c_i^{(0)})$ via local plane fitting to $\hat{\Omega}_+$
Initialize $\sigma_a = \sigma_b = 0.5$, $\sigma_c = 5$
for $t \leftarrow 1$ **to** $\#iterations$ **do**
    Add hypotheses from displets
    foreach *superpixel* $i \in \mathcal{S}$ **do**
        Add the current solution $\mathbf{n}_i^{(t-1)}$
        Sample the remaining half around the current solution:
        $a_i \sim \mathcal{N}(a_i^{t-1}, \sigma_a),\ b_i \sim \mathcal{N}(b_i^{t-1}, \sigma_b),\ c_i \sim \mathcal{N}(c_i^{t-1}, \sigma_c)$
        Sample the other half from the neighboring superpixels
    $(\mathbf{n}^{(t)}, \mathbf{d}^{(t)}) \leftarrow$ solve the discretized MRF using TRW-S (Kolmogorov, 2006)
    Update $\sigma_a = \sigma_b = 0.5 \times \exp(-(t-1)/10)$, $\sigma_c = 10 \times \exp(-(t-1)/10)$
Return $\mathbf{n}^{(t)}, \mathbf{d}^{(t)}$

4.3 Experiments

This section provides a thorough quantitative and qualitative analysis of the proposed displet model. As the number of images and objects per category in Middlebury (Scharstein and Szeliski, 2002) is too small to allow for a meaningful evaluation, we chose the more recent and challenging KITTI stereo dataset (Geiger *et al.*, 2012) as testbed for our experiments. Following the KITTI evaluation protocol, we perform all ablation studies

on the training set from which we randomly select 50 images for training the parameters in our model (Section 4.3.1). The remaining images are used for validation. In addition, we show convergence plots in terms of energy and error (Section 4.3.2). We submit our best performing configuration to the KITTI server for evaluation on the test set (Section 4.3.3). We perform block coordinate descent on the 50 training images to obtain the model parameters $\{\theta\}$ and $\{\tau\}$ which we fix throughout all our experiments (Section 4.3.4).

Semantic Segmentation: We manually annotated the training and test sets with pixel-wise car versus background labels and trained the associative hierarchical random fields model (Ladicky *et al.*, 2014a) for semantic segmentation. The algorithm is a multi-scale formulation of semantic segmentation by integrating information from different image scales (pixel, segment, and segment union/intersection). Ladicky *et al.* (2014a) show that the resulting hierarchical random fields model can be solved efficiently using graph-cut based move-making algorithms.

Image Features: In order to obtain sparse but high-quality input disparity maps, we process all stereo pairs using the semi-global matching framework (Hirschmüller, 2008) followed by a simple left-right consistency check to remove outliers. For calculating the matching costs, we use a combination of Census and Sobel features ("SGM", Yamaguchi *et al.* (2012)), as well as more recently proposed features based on convolutional neural networks ("CNN", Žbontar and LeCun (2016).

4.3.1 Ablation Study

Our first set of experiments conducted on the KITTI training set aims at assessing the contribution of each individual term in our energy. As we expect most notable improvements at reflective surfaces, we evaluate the error in all image regions (Table 4.1b) as well as the error only in reflective regions (Table 4.1a) using the KITTI ground truth. Unless otherwise stated, we report the percentage of outliers using the default outlier threshold of 3 pixels.

The first row in Table 4.1 shows the results of the input disparity maps, interpolated using the KITTI development kit. The following rows show our results when using only the unary term or combining it with one of the pairwise smoothness terms for superpixel boundaries ("Pair (Boundary)"), orientations ("Pair (Orientation)") and both ("Pair"), respectively. In combination, both smoothness terms are able to reduce the error by 16.6% for reflective and by 5.7% for all image regions considering the better performing CNN features. Adding the occlusion sensitive weight further reduces the error in all image regions but makes results slightly worse at reflective surfaces. This can be attributed to the fact that the input disparity maps contain holes and errors at reflective regions which are sometimes erroneously identified as occlusion boundaries hence lowering the smoothing

	CNN		SGM	
	Out-Noc	Out-All	Out-Noc	Out-All
Input $\hat{\Omega}$ (Interpolated)	19.84 %	22.98 %	22.60 %	25.52 %
Unary Only	17.72 %	21.72 %	19.38 %	23.36 %
Unary + Pair (Boundary)	15.96 %	19.67 %	16.18 %	19.94 %
Unary + Pair (Normal)	17.06 %	20.80 %	18.24 %	21.86 %
Unary + Pair	14.78 %	18.77 %	14.91 %	18.85 %
Unary + Pair + Occ	15.32 %	19.32 %	15.79 %	19.60 %
Unary + Pair + Disp	7.08 %	9.30 %	7.45 %	9.98 %
Unary + Pair + Occ + Disp	7.16 %	9.41 %	7.59 %	10.02 %

(a) **Reflective Regions**

	CNN		SGM	
	Out-Noc	Out-All	Out-Noc	Out-All
Input $\hat{\Omega}$ (Interpolated)	3.35 %	4.28 %	5.13 %	6.08 %
Unary Only	3.31 %	4.60 %	4.71 %	5.96 %
Unary + Pair (Boundary)	3.21 %	4.15 %	4.28 %	5.23 %
Unary + Pair (Normal)	3.28 %	4.31 %	4.52 %	5.60 %
Unary + Pair	3.12 %	3.95 %	4.13 %	4.93 %
Unary + Pair + Occ	3.04 %	3.88 %	4.07 %	4.80 %
Unary + Pair + Disp	2.87 %	3.64 %	3.87 %	4.57 %
Unary + Pair + Occ + Disp	2.78 %	3.55 %	3.76 %	4.50 %

(b) **All Regions**

Table 4.1: **Importance of Different Terms in the Model.** This table shows the performance of various model configurations on the validation set of KITTI for reflective regions (a) and for all regions (b) using the default error threshold of 3 pixels.

effect for these regions. Finally, adding the proposed displets to the model dramatically improves the results, reducing the number of outliers by additional 53.3% in reflective regions and by 8.6% in all regions.

Number of Proposals and Models: Next, we evaluate the influence of the number of object proposals as well as the variety of CAD models used for generating the displets. For these experiments, we focus our attention on the reflective regions as those are most affected by limiting the number of displets. As we obtain a different number of displet proposals in each image, we randomly draw subsets and plot the error with respect to the acceptance probability of a displet in Fig. 4.7 (left). Here, $P = 0$ corresponds to the case without displets and $P = 1$ corresponds to the case when making use of all available proposals for inference. The performance with respect to the number of CAD models used is shown in Fig. 4.7 (right). For this experiment, we randomly select a subset of models for each image (ranging from 0 to all 8 models) and discard all the proposals from

all other models. Both plots illustrate that reasonable results can be obtained with a small number of displet proposals and 3 models only. However, in both cases performance keeps increasing when adding more displets.

Number of Superpixels: In Fig. 4.8, we investigate the impact of the number of superpixels on the performance of our model. Similarly to Yamaguchi *et al.* (2012), we observe diminishing returns beyond 500 superpixels and chose 1000 superpixels as a reasonable trade-off between accuracy and performance for all other experiments.

4.3.2 Convergence

As our inference procedure is an iterative process with runtime linear in the number of iterations we plot the energy and errors through the iterations to determine the point of convergence. Fig. 4.9 shows the average energy and error over the images in the validation set using CNN (left) and SGM (right) as input disparity maps. As both the error and the energy stabilize after about 40 iterations, we set the maximum number of iterations to this value throughout all our experiments.

4.3.3 Results on the KITTI Stereo Benchmark

This section compares our performance with respect to the current state-of-the-art. We submitted our results for the KITTI test set using the best performing configuration according to Table 4.1b (CNN+Full model) to the KITTI evaluation server. As shown in Table 4.3, our method ranked first amongst more than 60 competitors in all evaluation categories at the time of submission. As expected, our improvements are particularly pronounced in reflective regions, but also improve overall performance even with respect to scene flow methods which take two or more stereo pairs of the sequence as input. The relatively weak performance of Ladicky *et al.* (2010) can be attributed to the simple semantic interaction model as well as the suboptimal graph-cuts based inference procedure.

Qualitative Results: Fig. 4.10 shows some qualitative results of our method without displets (left column in each sub-figure) as well as our full model including displets (right column in each subfigure). As evidenced by the error images in the last row of each subfigure, the proposed displet significantly reduces errors for the category car in a large number of images and even in extremely challenging scenarios such as the sub-figure in row 2, column 2. Two failure cases are highlighted at the bottom: In the left scene, errors on the caravan increase slightly as our collection of 3D CAD models doesn't contain an instance of this rather rare vehicle type. In the right scene the trunk of the car is extrapolated towards the building due to failures in the semantic segmentation (indicated in green) while the overall reconstruction of the trunk has improved.

4.3.4 Parameter Settings

Table 4.2 lists the parameter values we used throughout our experiments. The parameters in the first table are obtained using block coordinate descent on 50 randomly selected images from the KITTI training set. The values of the parameters in the table below have been determined empirically. Furthermore, Fig. 4.11 shows the sensitivity when varying the main parameters in our model for both CNN and SGM features. We vary one parameter at a time while setting all other parameters to their optimal values listed in Table 4.2. We observe that our method is not very sensitive with respect to small parameter changes for most of the parameters. The performance in reflective regions is highly dependent on the displet parameters as the induced sigmoid function defines the contribution of the inferred displets to the individual superpixels.

4.3.5 Runtime Analysis

Our MATLAB implementation with C++ wrappers requires on average 60 seconds for sampling the displets (parallelized using 12 cores), 5 seconds for initialization and 2.5 seconds for each of the 40 MP-PBP iterations, including graph construction (2.2 seconds) and TRW-S inference (0.3 seconds). In addition, semantic segmentation (Ladicky *et al.*, 2014a) requires 3.5 seconds per binary image segmentation and we obtain our initial disparity maps in 5 seconds for SGM or 100 seconds for CNN. We thus require 265 seconds in total for processing a single image using the full model in combination with CNN based matching costs.

Parameter	Value
Unary Threshold (τ_1)	6.98
Pairwise Boundary Threshold (τ_2)	3.40
Pairwise Normal Threshold (τ_3)	0.06
Pairwise Boundary Weight (θ_1)	1.50
Pairwise Normal Weight (θ_2)	586.87
Displet Weight (θ_3)	1.53
Occlusion Weight	1.00
Displet Consistency Weight (λ_θ)	6×10^4
Displet Consistency Sigmoid (λ_a, λ_b)	[7.90 0.82]

(a) **Optimized Parameters**

Parameter	Value
Number of Particles	30
Number of Superpixels	1000
Number of Iterations	40
TRWS - Number of Iterations	200
Particle Standard Deviations	[0.5 0.5 5]

(b) **Other Parameters**

Table 4.2: **Model Parameters.** Here, we list the values of the parameters in our model.

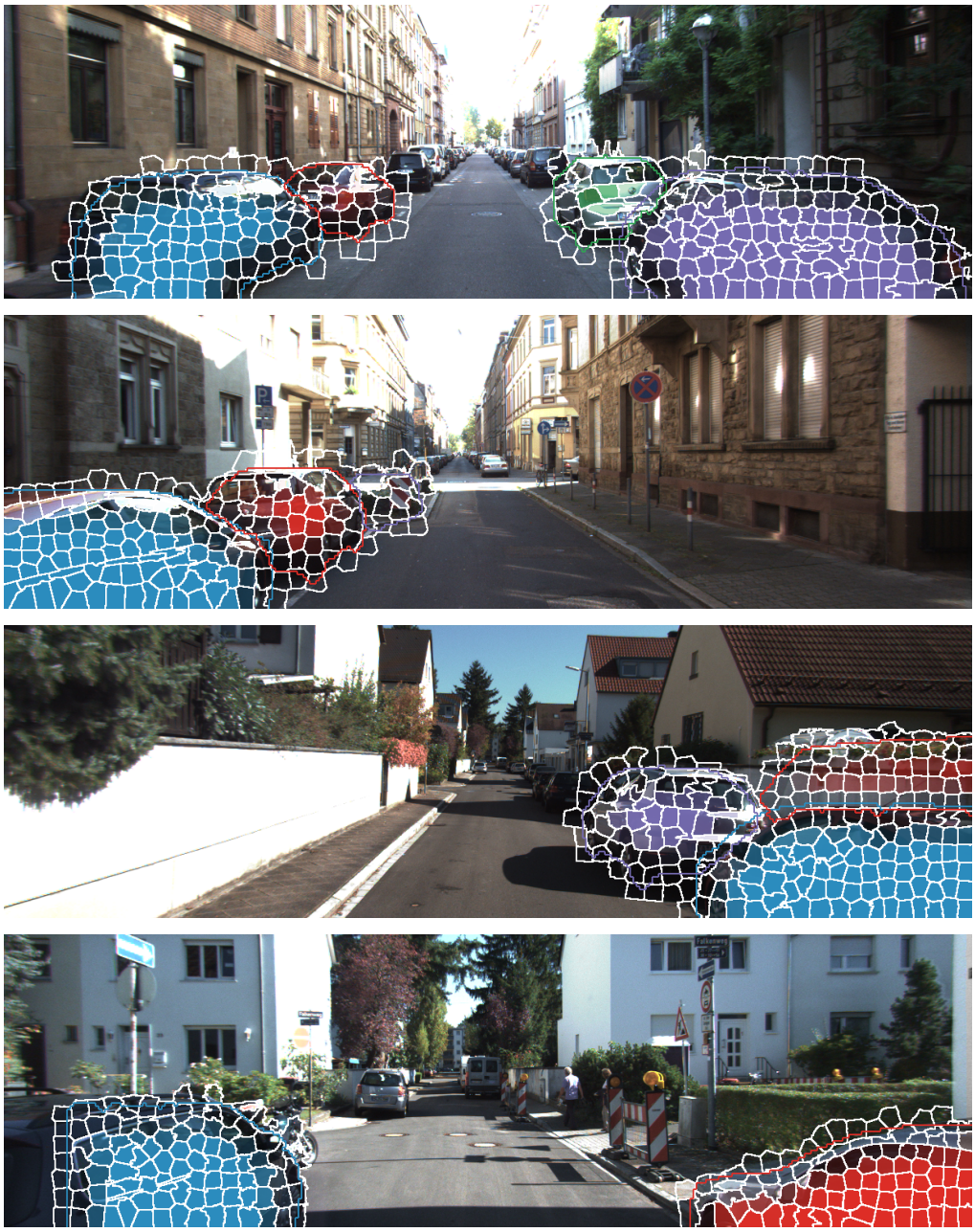
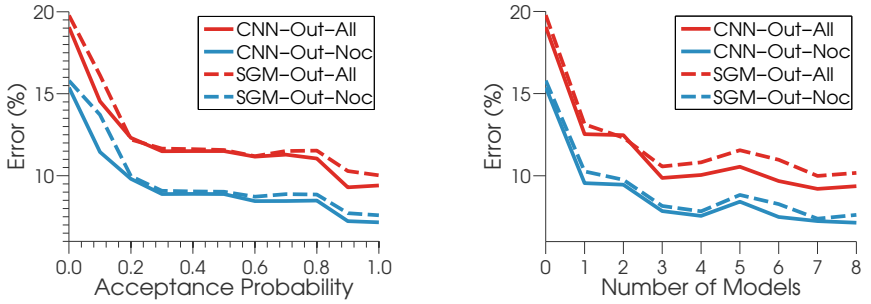
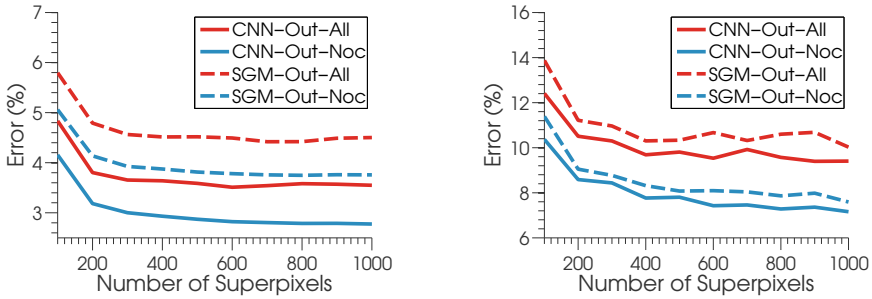


Figure 4.6: **Illustration of Displet Influence λ_{ki}.** We penalize superpixels which do not agree ($\mathbf{n}_i \neq \hat{\mathbf{n}}_{k,z_i}$) with active displets ($d_k = 1$). Less transparency indicates a higher penalty, i.e., we allow more deviation at the displet boundaries.



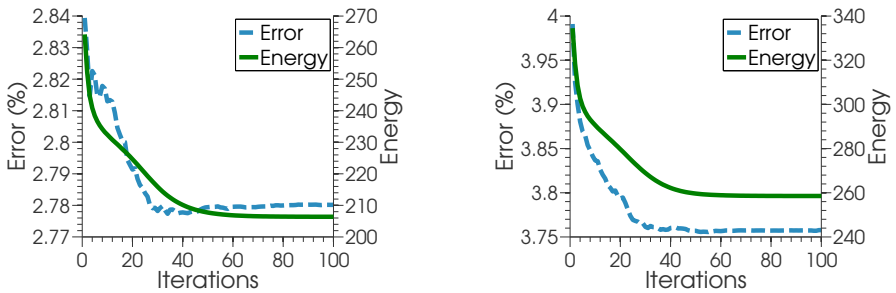
(a) Proposals (b) Models

Figure 4.7: **Number of Proposals and Models.** This figure shows the performance in reflective regions when limiting the number of object proposals (a) and models (b).



(a) All Regions (b) Reflective Regions

Figure 4.8: **Number of Superpixels.** This figure shows the performance when varying the number of superpixels for all regions ((a)) and for reflective regions (b).



(a) CNN (b) SGM

Figure 4.9: **Convergence of Error and Energy.** This figure shows the decrease in error and energy on all image regions vs. MP-PBP iterations using CNN (a) and SGM (b).

Rank	Method	Out-Noc	Out-All	Avg-Noc	Avg-All
1	Our Method	**8.40 %**	**9.89 %**	**1.9 px**	**2.3 px**
2	VC-SF * Vogel *et al.* (2014)	11.58 %	12.29 %	2.7 px	2.8 px
3	PCBP-SS Yamaguchi *et al.* (2013)	14.26 %	18.33 %	2.4 px	3.9 px
4	SPS-StFl * Yamaguchi *et al.* (2014)	14.74 %	18.00 %	2.9 px	3.6 px
5	CoP	15.30 %	19.15 %	2.7 px	4.1 px
6	SPS-St Yamaguchi *et al.* (2014)	16.05 %	19.34 %	3.1 px	3.6 px
7	DDS-SS Wei *et al.* (2014)	16.23 %	19.39 %	2.5 px	3.0 px
8	PCBP Yamaguchi *et al.* (2012)	16.28 %	20.22 %	2.8 px	4.4 px
9	PR-Sf+E * Vogel *et al.* (2013b)	17.85 %	20.82 %	3.3 px	4.0 px
10	StereoSLIC Yamaguchi *et al.* (2013)	18.22 %	21.60 %	2.8 px	3.6 px
11	MC-CNN Zbontar and LeCun (2014)	18.45 %	21.96 %	3.5 px	4.3 px
12	PR-Sceneflow * Vogel *et al.* (2013b)	19.22 %	22.07 %	3.3 px	4.0 px
⋮	⋮	⋮	⋮	⋮	
62	ALE-Stereo Ladicky *et al.* (2010)	83.80 %	84.37 %	24.6 px	25.4 px

(a) **Reflective Regions**

Rank	Method	Out-Noc	Out-All	Avg-Noc	Avg-All
1	Our Method	**2.47 %**	**3.27 %**	**0.7 px**	**0.9 px**
2	MC-CNN Zbontar and LeCun (2014)	2.61 %	3.84 %	0.8 px	1.0 px
3	SPS-StFl * Yamaguchi *et al.* (2014)	2.83 %	3.64 %	0.8 px	**0.9 px**
4	VC-SF * Vogel *et al.* (2014)	3.05 %	3.31 %	0.8 px	0.8 px
5	SPS-St Yamaguchi *et al.* (2014)	3.39 %	4.41 %	0.9 px	1.0 px
6	PCBP-SS Yamaguchi *et al.* (2013)	3.40 %	4.72 %	0.8 px	1.0 px
7	CoP	3.78 %	4.63 %	0.9 px	1.1 px
8	DDS-SS Wei *et al.* (2014)	3.83 %	4.59 %	0.9 px	1.0 px
9	StereoSLIC Yamaguchi *et al.* (2013)	3.92 %	5.11 %	0.9 px	1.0 px
10	PR-Sf+E * Vogel *et al.* (2013b)	4.02 %	4.87 %	0.9 px	1.0 px
11	PCBP Yamaguchi *et al.* (2012)	4.04 %	5.37 %	0.9 px	1.1 px
12	PR-Sceneflow * Vogel *et al.* (2013b)	4.36 %	5.22 %	0.9 px	1.1 px
⋮	⋮	⋮	⋮	⋮	⋮
62	ALE-Stereo Ladicky *et al.* (2010)	50.48 %	51.19 %	13.0 px	13.5 px

(b) **All Regions**

Table 4.3: **Quantitative Evaluation on the KITTI Stereo Benchmark.** This table shows the KITTI stereo leaderboards at time of submission using the default error threshold of 3 pixels. Evaluation is performed separately for reflective regions (a) and for all regions (b) of the KITTI test set. The numbers represent outliers (in %) and average disparity error (in pixels). Methods marked with an asterisk are scene flow methods which use two or more stereo image pairs as input.

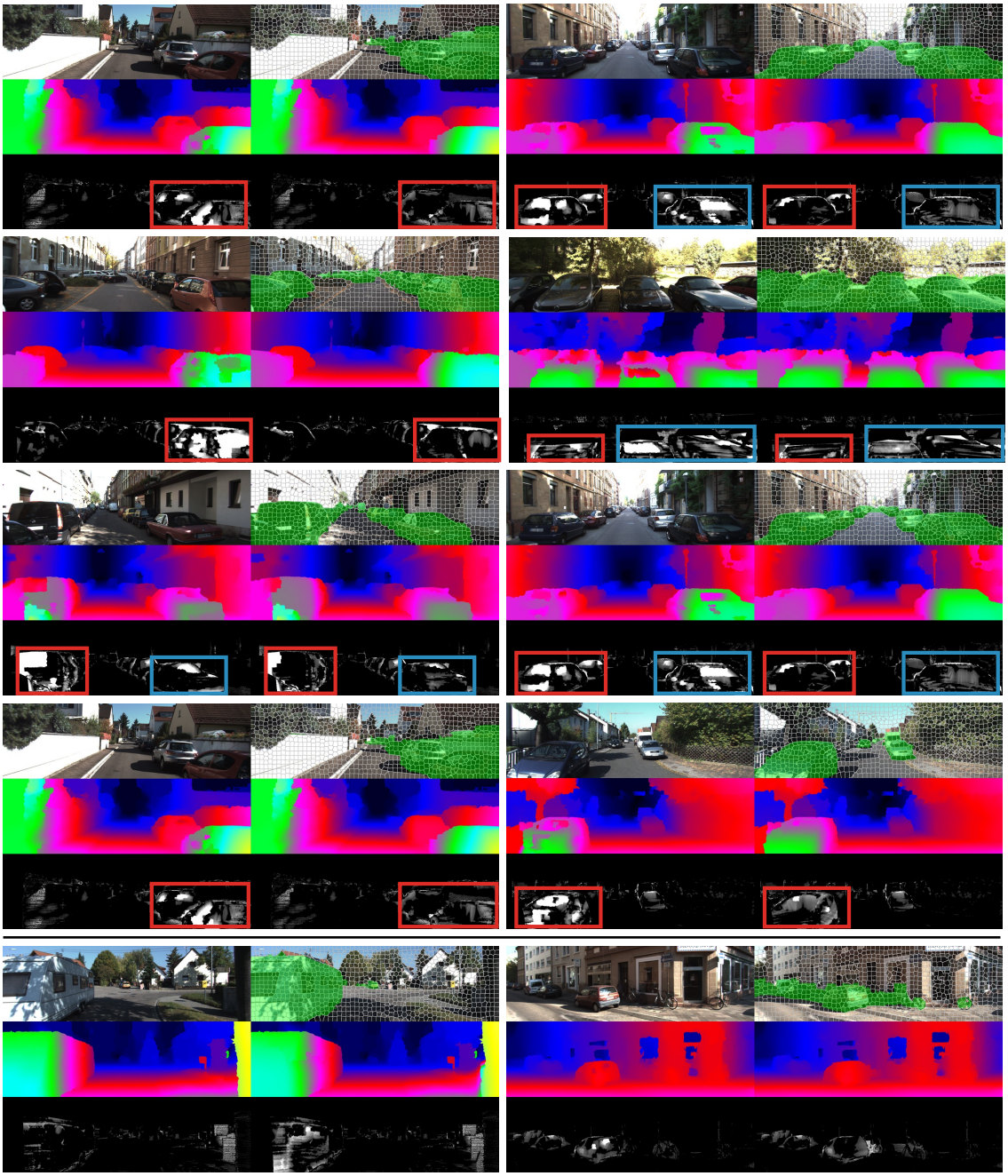
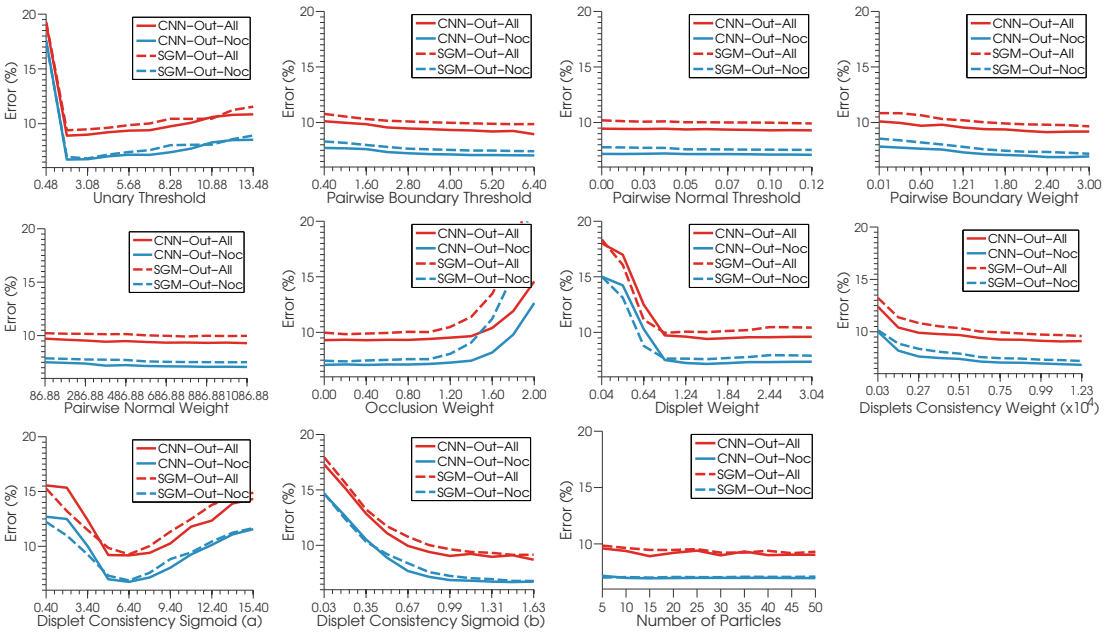
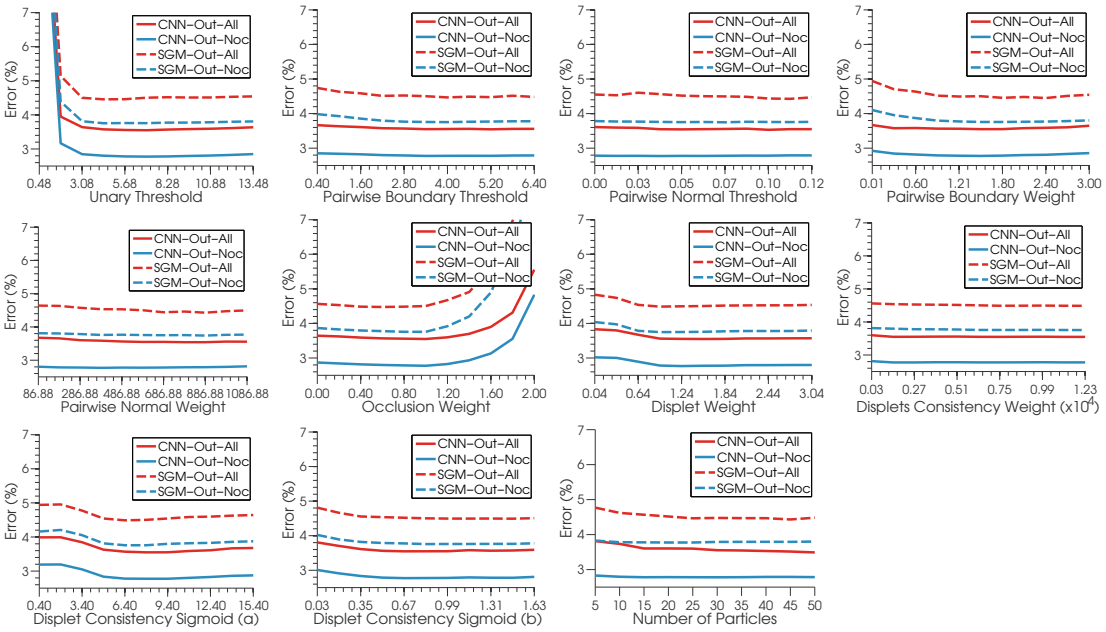


Figure 4.10: **Qualitiative Results.** Each subfigure shows from top-to-bottom: The input image, the superpixels and the semantic segments (colored green) which our method takes as input (first row), our inference results without and with displets (second row), and the corresponding error maps from ≤ 1 pixel error in black to ≥ 5 pixels error in white (third row). We mark regions with big improvements using red and blue rectangles. Two failure cases where the displets do not fit the data in terms of depth (bottom-left) and shape (bottom-right) are illustrated below the horizontal line.



(a) **Reflective Regions.** See text for details.



(b) **All Regions.** See text for details.

Figure 4.11: **Sensitivity to Parameters.** This figure shows the change in performance while varying one parameter at a time and keeping all others fixed at their optimal values. Errors are with respect to an error threshold of 3 pixels.

Chapter 5

Exploiting Object Similarity in 3D Reconstruction

According to a psychological study by Biederman (1987), a young child learns many categories from very few observations every day. This motivates many approaches in computer vision such as one-shot learning which aims to extract general knowledge from many samples of a few categories and utilize it for further categories with less samples in a more efficient way Fei-Fei *et al.* (2006). Inspired by the same observation, we exploit the shape similarities of objects for 3D reconstruction. As we drive long enough in a neighborhood recording sequences, we are very likely to see objects of similar type and shape such as cars and buildings. As we studied in the previews chapter for binocular stereo, cars are geometrically well-defined objects whose shape can be represented by a limited set of CAD models. Furthermore, man-made environments are designed to be visually pleasing, buildings also often exhibit similar shapes in local neighborhoods. In this chapter, we jointly learn a shape model for frequently occurring objects on the street scenes, while reconstructing the scenes.

Our approach is summarized as follows: We first obtain an initial 3D reconstruction by structure from motion and volumetric fusion of disparity maps using a memory efficient representation based on voxel hashing (Nießner *et al.*, 2013). Next, we train several generic 3D object detectors by extending exemplar SVMs to the 3D representation. Given the 3D box proposals from these detectors, we formulate a joint objective function over alignment of these boxes, learning volumetric 3D shape models of objects and assignment of boxes to learned shape models. We model the shape of objects using low dimensional linear embeddings, i.e. PCA, therefore the latent shape variables as well as the shape model parameters are inferred in the joint optimization. This formulation results in a discrete-continuous optimization problem which we solve using block coordinate descent.

Joint reconstruction and shape learning can be useful in many ways. First of all, the reconstruction of outdoor scenes can be very challenging with many regions which cannot be reconstructed due to the missing evidence in the images. By regularizing across shapes, we transfer surface knowledge for the missing surfaces and observe very clear improvements in the completeness of the reconstruction. For the regularization, we ad-

(a) Fisheye Image (b) Lighting (c) Occlusions

(d) Saturation (e) Reflections (f) Appearance

Figure 5.1: **Challenges of 3D Reconstruction from Movable Platforms.** We have collected a suburban dataset for omnidirectional 3D reconstruction from fisheye images (a). Our dataset has been captured under normal daylight conditions and poses a variety of challenges to current reconstruction pipelines, including uncontrolled light conditions (b), occlusions (c), sensor saturation at bright surfaces (d), reflecting surfaces (e), and large appearance changes between successive frames when driving at regular speeds (f).

here to the principle of parsimony by limiting the number of models and parameters. Second, by clustering objects according to their shape, we obtain a decomposition of a scene into its constituent objects in terms of 3D bounding boxes. Finally, other tasks such as recognition or synthesis could benefit from the learned 3D shape models.

In the following, we first introduce the data we recorded from fisheye cameras and show the potential challenges for the algorithms in Section 5.1. Then, we explain the structure from motion pipeline for obtaining camera parameters and poses in Section 5.2. We continue with the volumetric fusion based on voxel hashing for the initial 3D reconstruction in Section 5.3. Given the 3D box proposals from generic 3D object detectors (Section 5.4), we formulate the joint optimization in Section 5.5 and explain the details of inference in Section 5.6. In Section 5.7, we evaluate the proposed approach by comparing with respect to the initial reconstruction from Section 5.3 as well as to a set of baselines in terms of both the accuracy and the completeness of reconstructions.

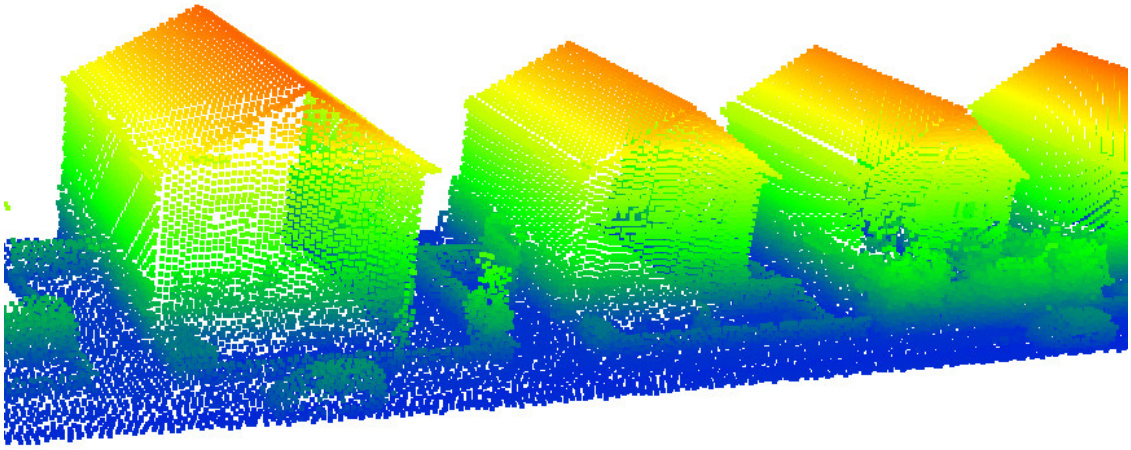


Figure 5.2: **Ground-Truth Point Cloud.** This figure shows our ground-truth 3D point cloud captured by the laser scanners and resampled uniformly for one part of the sequence.

5.1 Data and Challenges

The existing datasets for multi-view reconstruction (e.g., Middlebury Seitz *et al.* (2006)) focus on small scenes of single objects. To demonstrate our model's capabilities in large scenes with multiple objects, we have collected a suburban dataset for omnidirectional 3D reconstruction from fisheye images. In contrast to Middlebury Seitz *et al.* (2006), sequences have been recorded under regular daylight conditions. Therefore, the dataset poses various challenges for the reconstruction pipelines. As illustrated in Fig. 5.1, lighting conditions are difficult, occlusions are omnipresent and many objects are only visible in a few frames. These cases are shown to be very challenging for the existing approaches in our evaluations (Section 5.7). We propose to overcome these challenges by modeling the shape similarities present on the street scenes.

We have recorded our sequences from a moving platform by driving in areas where objects of similar shape such as buildings and cars occur frequently. For recording, we have equipped a station wagon with fisheye cameras to the side as well as a Velodyne and a pushbroom laser scanner in order to generate ground truth. All cameras and laser scanners have been synchronized at a frame rate of roughly 10 fps, yielding about one captured frame every meter at typical driving speeds of 25 mph. Using this setup, we recorded a sequence of 320 frames containing several buildings and cars. For evaluation, we subsample the laser scanner point cloud and the meshes produced by the methods equidistantly at 0.1 m. As the roof and the side walls of buildings are sometimes sampled very sparsely by the laser scanner, we manually extruded incomplete planar surfaces and resampled points uniformly. See Fig. 5.2 for an illustration of our ground-truth 3D point cloud.

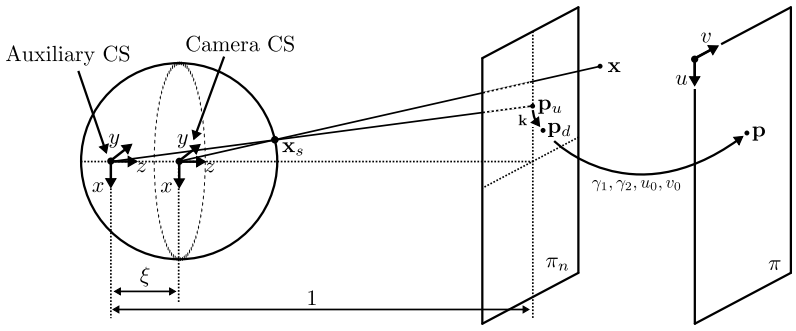


Figure 5.3: **Illustration of the Projection Model**. This figure illustrates the projection model assumed for the fiseye cameras as proposed in Mei and Rives (2007).

5.2 Structure from Motion

Given a sequence of fisheye images, we first obtain the intrinsic camera parameters as well as the vehicle and camera poses using the approach of Heng *et al.* (2013). We rectify the images of adjacent frames spherically such that epipolar lines become horizontal and run semi-global matching Hirschmüller (2008) in order to obtain disparity maps. We explain and provide the details for each of these steps in the following.

5.2.1 CamOdoCal

CamOdoCal[1] is a calibration toolbox for automatic intrinsic and extrinsic calibration of a multi-camera rig mounted on a moving vehicle Heng *et al.* (2013); Lee *et al.* (2013); Heng *et al.* (2014, 2015). CamOdoCal makes use of the wheel odometry based on wheel's motion which is readily available as a strong prior for camera pose estimation in a highly optimized and automated pipeline.

The goal of the intrinsic calibration is to obtain the optimal parameters for an assumed camera projection model which relates the 2D image points to the 3D points in the world. The optimality is defined in terms of the reprojection error. The goal of the extrinsic calibration is to find the camera poses relative to a reference frame on the vehicle, typically the odometry frame. At the end of a successful calibration process, feature points can be accurately reprojected between different cameras and the odometry. Next, we explain the details of intrinsic and extrinsic calibration.

Instrinsic Calibration

As the camera projection model, we use unified projection model of Mei and Rives (2007) generalized to fisheye cameras. This projection model defines the mapping which takes a 3D point in camera coordinates and projects it to a 2D point on the image plane.

[1] `https://github.com/hengli/camodocal`

Let $\mathbf{x} = (x, y, z)^T$ denote a 3D point in camera coordinates and let $\mathbf{p} = (u, v)^T$ denote a pixel in the image plane as illustrated in Fig. 5.3. We first project the 3D world to a point $\mathbf{x}_s = (x_s, y_s, z_s)^T$ onto the unit sphere

$$\mathbf{x}_s = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \tag{5.1}$$

followed by a translation and a projection onto the normalized plane π_n:

$$\mathbf{p}_u = \left(\frac{x_s}{z_s + \xi}, \frac{y_s}{z_s + \xi} \right)^T \tag{5.2}$$

Next, the undistorted point $\mathbf{p}_u = (u_u, v_u)^T$ is mapped to a distorted point $\mathbf{p}_d = (u_d, v_d)^T$ by applying radial and tangential distortion

$$\mathbf{p}_d = (1 + k_1\rho^2 + k_2\rho^4 + k_5\rho^6)\mathbf{p}_u + \begin{pmatrix} 2k_3 u_u v_u + k_4(\rho^2 + 2u_u^2) \\ 2k_4 u_u v_u + k_3(\rho^2 + 2v_u^2) \end{pmatrix} \tag{5.3}$$

where $\rho = \sqrt{u_u^2 + v_u^2}$. The final projection of $\mathbf{p}_d = (u_d, v_d)^T$ to a pixel $\mathbf{p} = (u, v)^T$ on the image plane π follows a standard pinhole camera model

$$\mathbf{p} = \begin{pmatrix} \gamma_1 u_d + u_0 \\ \gamma_2 v_d + v_0 \end{pmatrix} \tag{5.4}$$

The intrinsic camera model parameters are:

- ξ: Displacement of auxiliary coordinate system
- $\mathbf{k} = (k_1, k_2, k_3, k_4, k_5)^T$: Distortion parameters
- $\gamma = (\gamma_1, \gamma_2)^T$: Generalized focal length
- $\mathbf{c} = (u_0, v_0)^T$: Principal point

The projection of a 3D point to the image plane can be defined as $\pi : \mathbf{x} \mapsto \mathbf{p}$. By applying these transformations in the reverse order, a point $\mathbf{p}$ on the image plane π can be lifted to the corresponding 3D point on the sphere $\mathbf{x}_s$, i.e., $\pi^{-1} : \mathbf{p} \mapsto \mathbf{x}_s$. However, the inverse distortion model has no analytic solution and requires numerical computations in iterations (Hartley and Zisserman, 2004). Fig. 5.3 illustrates the projection model. During calibration, using a chessboard with equal known dimension, corners are detected and the parameters of the model is estimated and refined in a non-linear optimization scheme for minimizing reprojection error.

Extrinsic Calibration

We summarize the steps taken in the extrinsic calibration stage. For details of each step, please refer to Heng *et al.* (2013); Lee *et al.* (2013).

- **Monocular Visual Odometry (VO)**: The goal of this step is to obtain the sets of camera poses and inlier feature tracks for each camera. First, feature points and their descriptors are extracted. Then, in iterations, the Perspective-Three-Point (P3P) method with RANSAC (Kneip *et al.*, 2011) is used to determine the camera pose and identify inlier feature tracks, followed by a sliding window bundle adjustment.

- **Camera-Odometry Calibration**: Given the initial camera poses and the odometry data, the camera-odometry calibration is performed as proposed by Heng *et al.* (2013) in order to obtain an initial estimate of the camera-odometry transforms from an arbitrary number of segments of camera poses.

- **The Scene Point Reconstruction**: The inlier feature correspondences from monocular VO step are triangulated using the odometry data and the initial estimate of the camera-odometry transforms.

- **Bundle Adjustment**: The camera-odometry transforms and 3D scene points are optimized to minimize the reprojection error across all frames for all cameras while keeping the vehicle poses fixed. This step ensures optimal camera-odometry transforms that are consistent with each other.

- **Loop Closure Detection**: For global consistency, loop closures are detected by using a vocabulary tree (Gálvez-López and Tardós, 2012).

- **Robust Pose Graph Optimizaiton**: Wrong loop closures from the previous step are identified and removed by using the EM algorithm using the Bayesian formulation proposed in Lee *et al.* (2013).

- **Inter-Camera Consistency**: In order to achieve sub-pixel accuracy in the reprojection between cameras, feature point correspondences are established as constraints across different cameras.

- **Final Bundle Adjustment**: In a final bundle adjustment step, all the intrinsics, extrinsics, odometry poses and the 3D scene points are optimized for a metrically accurate and globally consistent calibration.

- **GPS Constraint**: We introduce a further constraint to regularize against the GPS data for accurate georegistered camera poses.

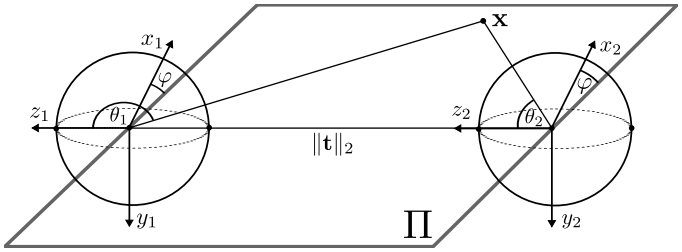


Figure 5.4: **Spherical Rectification.** Given two fisheye cameras (x_1, y_1, z_1) and (x_2, y_2, z_2), we rectify the images yielding an angular representation (θ, φ) where pixels in the same row (i.e., same φ) are located on an epipolar plane Π.

5.2.2 Spherical Rectification

Before stereo matching, we rectify the images such that epipolar lines are straight and horizontal. This is achieved by spherically rectifying the two images according to the motion between the cameras. A pixel in a spherically rectified image corresponds to the inclination angle (θ, u-coordinate) and azimuth angle (ϕ, v-coordinate) of the respective viewing ray in the rectified coordinate system.

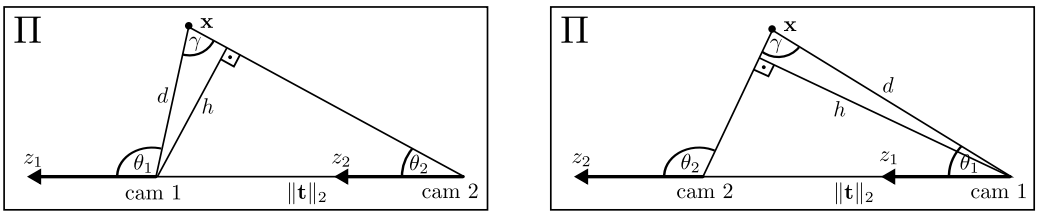
More formally, let $\mathbf{H} \in \mathbb{R}^{3\times4}$ denote the rigid body transformation which maps a point in the coordinate system of the reference camera, to the coordinate system of the other camera. Let further $\mathbf{R}_1 \in \mathbb{R}^{3\times3}$and $\mathbf{R}_2 \in \mathbb{R}^{3\times3}$ denote the rectifying rotations which represent the rectified camera coordinate system with respect to the original camera coordinate system, respectively. $\mathbf{R}_1$ and $\mathbf{R}_2$ are obtained from $\mathbf{H}$ in such a way that the z-axis of $\mathbf{R}_1$ and $\mathbf{R}_2$ aligns with the ray passing through all four epipoles $e_{11}, e_{12}, e_{21}, e_{22}$ and $\mathbf{R}_2 = \mathbf{R}\mathbf{R}_1$ where $\mathbf{R}$ is the rotational component of $\mathbf{H}$. The coordinate systems of the first and second camera after spherical rectification are illustrated in Fig. 5.4. Here $\mathbf{t}$ denotes the translational component of $\mathbf{H}$ and Π denotes the epipolar plane. Note how the epipolar rectification ensures that the azimuth angle φ for a world point $\mathbf{x}$ is the same in both cameras.

Mapping: Rectified → Unrectified

Let $(\theta_{min}, \theta_{max}, \theta_{num})$ and $(\varphi_{min}, \varphi_{max}, \varphi_{num})$ denote the parameters of the rectified images, where $\theta \in [\theta_{min}, \theta_{max})$, $\varphi \in [\varphi_{min}, \varphi_{max})$ denote the angular ranges and $\theta_{num}, \varphi_{num}$ denote the width and height of the rectified images. For 180° fisheye cameras, the maximum range is $\theta \in [0, \pi]$, $\varphi \in [-\pi/2, \pi/2]$. A point $\mathbf{p}_r$ on the rectified image plane π_r can be projected to a point $\mathbf{p}$ on the image plane π as follows: We first obtain θ, φ from the rectified pixel coordinates $\mathbf{p}_r = (u_r, v_r)^T$

$$\theta = \theta_{min} + u_r * (\theta_{max} - \theta_{min})/\theta_n \tag{5.5}$$

$$\varphi = \varphi min + u_r * (\varphi_{max} - \varphi_{min})/\varphi_n \tag{5.6}$$



(a) Negative z Direction ($1 \rightarrow 2$) (b) Positive z Direction ($1 \rightarrow 2$)

Figure 5.5: **Illustration of Reconstructing a 3D Point.** This figure illustrates the two cases for reconstructing a 3D point $\mathbf{x}$, namely camera moving into the direction of negative z (a) and positive z (b).

and compute the corresponding point on the unit sphere as

$$\mathbf{x}_s = \mathbf{R}_i \begin{pmatrix} \sin(\theta)\cos(\varphi) \\ \sin(\theta)\sin(\varphi) \\ \cos(\theta) \end{pmatrix} \tag{5.7}$$

where $\mathbf{R}_i$ is the rectifying rotation corresponding to the respective camera. The mapping into the image plane π is realized using the projection model described in Section 5.2.1: $\mathbf{p} = \pi(\mathbf{x}_s)$.

Mapping: Unrectified $\rightarrow$ Rectified

Similarly, a point $\mathbf{p}$ on the image plane π can be projected to a point $\mathbf{p}_r$ on the rectified image plane π_r. We first obtain $\mathbf{x}_s$ by lifting $\mathbf{p}$ to the unit sphere and rotating it into the rectified coordinate system

$$\mathbf{x}_r = \mathbf{R}_i^{-1}\mathbf{x}_s \tag{5.8}$$
$$\mathbf{x}_s = \pi^{-1}(\mathbf{p}) \tag{5.9}$$

where, $\mathbf{R}_i$ denotes the rectifying rotation corresponding to the respective camera. The angles φ and θ can be extracted from the rotated point on the sphere $\mathbf{x}_r = (x_r, y_r, z_r)^T$

$$\theta = \arccos(z_r) \tag{5.10}$$
$$\varphi = \arcsin(y_r / \sin(\theta)) \tag{5.11}$$

and we obtain $\mathbf{p}_r$ as:

$$\mathbf{p}_r = \begin{pmatrix} (\theta - \theta_{min})\theta_n/(\theta_{max} - \theta_{min}) \\ (\varphi - \varphi_{min})\varphi_n/(\varphi_{max} - \varphi_{min}) \end{pmatrix} \tag{5.12}$$

5.2.3 Depth Computation and Triangulation

We run semi-global matching Hirschmüller (2008) on the rectified stereo pairs in order to obtain disparity maps. Given a pixel in a rectified image of the first camera and a corresponding disparity map, a 3D point $\mathbf{x}$ is reconstructed in the coordinate system of the first camera using Eq. 5.7:

$$\mathbf{x} = d(\theta_1, \gamma)\mathbf{R}_1 \begin{pmatrix} \sin(\theta_1)\cos(\varphi) \\ \sin(\theta_1)\sin(\varphi) \\ \cos(\theta_1) \end{pmatrix} \tag{5.13}$$

where $\mathbf{R}_1$ is the rectifying rotation of the first camera and $d(\theta_1, \gamma)$ denotes the distance of $\mathbf{x}$ from the camera and depends on the inclination angle θ_1 and the unsigned angular disparity $\gamma = |\theta_1 - \theta_2|$. For calculating $d(\theta_1, \gamma)$, two cases have to be distinguished, as illustrated in Fig. 5.5: The case where the camera translates into the direction of $-z$ and the case where the camera translates into the direction of z. In the first case (Fig. 5.5a), we have

$$\sin\gamma = \frac{h}{d} \qquad \sin\theta_2 = \frac{h}{\|t\|_2} \qquad \theta_2 = \theta_1 - \gamma$$

$$d(\theta_1, \gamma) = \frac{h}{\sin\gamma} = \frac{\sin(\theta_1 - \gamma)\|t\|_2}{\sin\gamma} \tag{5.14}$$

Similarly, for the second case (Fig. 5.5b), we obtain

$$d(\theta_1, \gamma) = \frac{\sin(\pi - \theta_1 - \gamma)\|t\|_2}{\sin\gamma} \tag{5.15}$$

In this section, we first described how to calibrate fisheye cameras in a multi-camera rig in order to obtain the intrinsic and extrinsic camera parameters. Given these parameters, we can define the mappings from rectified images to unrectified images and vice versa. After spherically rectifying fisheye images and estimating depth maps on rectified stereo pairs, 3D points can be triangulated using the rectifying rotation and depth estimates. Next, we explain the volumetric fusion of depth maps using these operations.

5.3 Volumetric Fusion

This section describes our basic volumetric reconstruction pipeline which serves as input and baseline to our method. We leverage the truncated signed distance function (TSDF) representation by Curless and Levoy (1996). Among the surface reconstruction algorithms, TSDF falls into the class of structured implicit functions that use samples of a continuous function to combine the structured data. For efficiency, we leverage voxel hashing proposed by Nießner *et al.* (2013). In the TSDF representation, surfaces correspond to the 0-level set of the TSDF which can be obtained by using the marching cubes

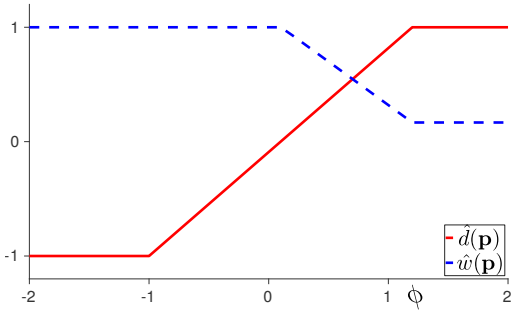


Figure 5.6: **TSDF Illustration.** This figure, adapted from Steinbrucker *et al.* (2013), visualizes the truncated signed distance function and the linear weighting function with respect to the distance from surface. The measured point in the disparity estimation is reconstructed and truncated using the truncation threshold ϕ to obtain the TSDF value at the voxel center, $\hat{d}(\mathbf{p})$, and the corresponding weight $\hat{w}(\mathbf{p})$ is determined based on that value using the Eq. 5.18.

algorithm (Lorensen and Cline, 1987). In the following, we introduce the TSDF representation (Section 5.3.1) and explain the details of voxel hashing (Section 5.3.1) and surface extraction (Section 5.3.2).

5.3.1 Representation

TSDF provides a general volumetric representation without making any assumptions about the 3D surface topology which is unknown in our case. The directional uncertainty is represented through a cumulative weighted signed distance function. It is based on incremental updates by integrating one disparity image at a time. Each disparity image is first converted to a distance function and then combined with the already acquired volume using a simple additive scheme:

$$d_{i+1}(\mathbf{p}) = \frac{w_i(\mathbf{p})d_i(\mathbf{p}) + \hat{w}(\mathbf{p})\hat{d}(\mathbf{p})}{w_i(\mathbf{p}) + \hat{w}(\mathbf{p})} \tag{5.16}$$

$$w_{i+1}(\mathbf{p}) = w_i(\mathbf{p}) + \hat{w}(\mathbf{p}) \tag{5.17}$$

where $\mathbf{p} \in \mathbb{R}^3$ denotes the location of a cell in the volumetric grid, and $\hat{d}(\cdot)$, $\hat{w}(\cdot)$ represent the truncated signed distance value and weight of the current observation (i.e., disparity map), respectively. The truncation ensures that surfaces on opposite sides of the objects do not interfere with each other. In addition, the truncation provides compactness and efficiency in updating the volume. Weights represent uncertainty due to the variations across disparity maps and they are set to 0 in unobserved regions and outside the truncation band and 1 close to the surface. Following the definition proposed in Steinbrucker *et al.* (2013), we use a weight function that slowly decays behind the surface as illustrated

in Fig. 5.6:

$$\hat{w}(\mathbf{p}) = \begin{cases} 1 & \text{if } \hat{d}(\mathbf{p}) < \delta \\ \frac{\phi - \hat{d}(\mathbf{p})}{\phi - \delta} & \text{if } \hat{d}(\mathbf{p}) \geq \delta \text{ and } \hat{d}(\mathbf{p}) \leq \phi \\ 0 & \text{if } \hat{d}(\mathbf{p}) > \phi \end{cases} \tag{5.18}$$

where the truncation threshold ϕ is set to 1.2 m and the other threshold δ to 0.005. As can be seen, TSDF is order independent, i.e. results are not biased by earlier scans and furthermore, incremental updates allow for straightforward parallelization.

As described in Curless and Levoy (1996), the algorithm based on regular voxel grid can be summarized as follows: By iterating through each disparity map, triangles are constructed over 2×2 blocks on the disparity map. Depth discontinuities are avoided by discarding triangles with edge lengths that exceed a threshold. After obtaining the triangle mesh, the next step is updating the voxel grid. The signed distance values are computed by casting a ray from the sensor through each voxel near the surface and then intersecting it with the triangle mesh. The weight for each point is computed by bilinear interpolation of the weights at the intersecting triangle vertices. After computing the signed distance and the weight values, the voxel grid can be updated using Eq. 5.16 and Eq. 5.17.

Slanted Surfaces: As we found that approximating the distances $\hat{d}(\cdot)$ by calculating them along the viewing ray (Steinbrucker *et al.*, 2013) results in artifacts when the ray hits the surface at highly slanted angles, we use a different strategy which approximates the TSDF values more faithfully: We triangulate each disparity map and convert the resulting meshes into volumes using a 3D signed distance transform[2]. For each triangle, first a range of affected grid voxels are collected in an *update list*. Then, for each voxel in the *update list*, we compute its distance to the triangle depending on whether the closest point is on the surface, or the triangle edges, or triangle vertices. If the computed distance is smaller than the current TSDF value stored in that voxel, the distance is updated. This leads to correct distance estimates also for surfaces which are not fronto-parallel.

Voxel Hashing

The original approach of Curless and Levoy (1996) based on a regular voxel grid, does not scale well due to the increasing memory requirement. In order to handle large-scale reconstructions efficiently, we leverage the voxel hashing of Nießner *et al.* (2013).

The TSDF representation is already optimized to some extent through truncation, i.e. by storing the signed distance in a region around the surface. In other words, updates to estimate the surface are limited to these regions. Furthermore, voxels outside the truncation region are marked as free-space. Nießner *et al.* (2013) state that the majority of

[2] http://grail.cs.washington.edu/software-data/ply2vri/

data stored in the regular voxel grid is marked either as free space or as unobserved space rather than as surface data. Based on this observation, they propose to use a spatial hashing technique to exploit the underlying sparsity in the TSDF representation and reduce the memory requirement. The hashmap representation allows to compactly store, access and update the TSDF representation through efficient look-up of voxel blocks. It also supports dynamic allocations and updates to handle the unknown and continually changing geometry. The potential collisions which are common to every hashing operation are reduced and resolved through an offset pointer.

The first functionality is the integration of a new depth map into the existing TSDF representation. For each estimate on the depth image, we update its corresponding 3D voxels within a neighborhood. The voxels to be updated are pushed into a queue and then all the voxels affected are updated at once. The update is performed by back-projecting the voxels to the depth map and computing the distance function. This way, a voxel is updated only once while fusing a depth map. For each grid point, the distance value is the weighted average of grid points in a 8^3 neighborhood. The other functionalities are inserting and retrieving a TSDF value. The hashing function H is defined based on *xor* operations between the integer rounded world coordinates x, y, z multiplied by large prime numbers p_1, p_2, p_3, respectively:

$$H(x,y,z) = (x \cdot p_1) \oplus (y \cdot p_2) \oplus (z \cdot p_3) \ \ mod\, n \tag{5.19}$$

Here, $\oplus$ denotes the *xor* operation and n is the hash table size. Lastly, we also store the information necessary for the conversion between the world and the voxel coordinates.

5.3.2 Surface Extraction

In the TSDF representation, surfaces are implicitly represented as the 0-level set of the TSDF function. Eq. 5.16 represents the signed distance of each point p to the nearest surface along the viewing ray. After constructing these functions on a discrete voxel grid, surfaces can be recovered by extracting iso-surfaces corresponding to $d(\mathbf{p}) = 0$.

The final surface is typically obtained using the marching cubes algorithm by Lorensen and Cline (1987). The marching cubes algorithm is based on traversing the volume and mapping each voxel to a pre-defined configuration to locate the surface inside the voxel. Each configuration defines a way the surface can intersect a cube or voxel. Each vertex of the cube is represented by an on or off bit to represent that it is inside or outside the surface, respectively. This results in $2^8 = 256$ configurations which is reduced to a smaller number of unique cases through equivalence in topology and rotational symmetries. Each index corresponds to a list of edge intersections which describe how a surface cuts through the cube. The surface intersection along an edge can be linearly interpolated using the values at the vertices that are connected by the edge. The overall surface representation is obtained by connecting all the triangulated surfaces from all the cubes.

Hole Filling: Holes in the reconstruction are unavoidable due to unobserved surfaces in the scene. Ideally, the algorithm should automatically fill these holes with plausible surfaces, at least for some simple cases. The original Curless and Levoy (1996) uses a method called space carving to fill the holes. Space carving classifies all points in the volume into one of three states: unseen, empty, or near the surface. The lines in front of the observed surface are marked empty while the regions behind remain unseen. Holes correspond to the regions between unseen and empty regions. Besides the zero-crossings of the signed distance function, additional surfaces are introduced in the hole regions. Space carving is guaranteed to produce watertight surfaces, however, it may also lead to implausible surfaces due to the redundant bridging. Davis *et al.* (2002) propose a heat diffusion procedure to smoothly extend the observed surfaces. The algorithm iteratively smooths the TSDF values to propagate information from the observed surfaces to unobserved grid points, eventually spanning across the holes. In our experiments, we found that the smoothing leads to surfaces in small hole regions that generally blend well with the observed surface.

5.4 Object Detection

Given the volumetric 3D reconstruction from the previous section, we are interested in discovering objects of similar 3D shape. For that purpose, we train 3D object detectors on a small subsequence annotated with 3D boxes for the objects of interest, i.e. building and car. Before joint reconstruction, we run these detectors on the initial reconstruction to find instances of objects. The output of this step is a set of 3D bounding boxes which form the basis for our joint optimization in Section 5.5. Detection of objects in 3D space has several advantages over the detection in the image domain: Besides the fact that similar 3D shapes can be matched even if the appearance disagrees, 3D models must not be scale invariant, thereby gaining robustness. A variety of methods can be leveraged for object discovery in 3D, including methods based on convexity/symmetry criteria (Karpathy *et al.*, 2013), discriminative approaches (Zhang *et al.*, 2014b) or graph matching (Zhang *et al.*, 2014c).

In this work, we use a discriminative approach to obtain 3D box proposals: For each object category, we train an ensemble of linear exemplar SVMs (Malisiewicz *et al.*, 2011) directly on the TSDF volume using a small (≤ 3) number of annotated instances. Compared to learning more complex features (Song and Xiao, 2014), we found this simple approach sufficient for our needs. First, we extract a small subsequence for training in which we label the relevant objects using 3D bounding boxes. For efficient object labeling we developed a 3D visualization tool based on WebGL, which allows to label objects in 3D point clouds in a few seconds. Next, we define a feature vector by concatenating all voxels within the 3D box. Each voxel comprises the truncated signed distance to the surface $d(\mathbf{p})$ as well as a binary flag indicating if the voxel has been observed or not (i.e., it is within the truncation limit) as feature. We train exemplar SVMs on this

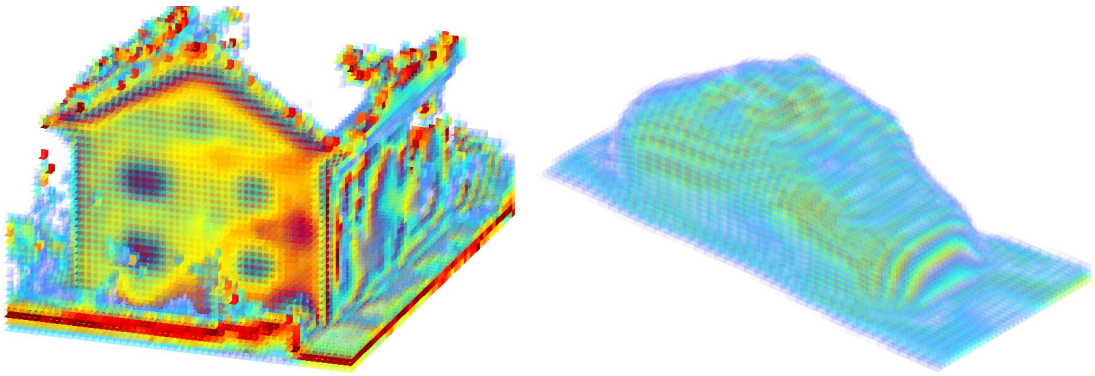


Figure 5.7: **Visualization of TSDF Model.** This figure shows a volumetric visualization of the learned TSDF model mean for a building (left) and a car (right). Red colors indicate voxels close to the estimated surface (i.e., 0-level set)

representation using several rounds of hard negative mining and apply them in a sliding window fashion to the full 3D reconstruction. As 3D representations offer the advantage of scale invariance, we slide and rotate a 3D box of fixed scale over the 3D reconstruction volume. Fig. 5.8 (left) illustrates our detection results.

5.5 Joint Object Reconstruction

Given the initial volumetric 3D reconstruction and the bounding box proposals, our goal is to jointly recover objects with similar shape while learning an object model for each category. More formally, let $d(\mathbf{p}) \in \mathbb{R}$ denote the truncated signed distance to the closest surface at point $\mathbf{p} \in \mathbb{R}^3$. Let further $w(\mathbf{p}) \in \mathbb{R}_{\geq 0}$ denote the weight at point $\mathbf{p}$ which takes $w = 0$ in unobserved regions or outside the truncation band, and $w = 1$ close to the estimated surface. As the TSDF representation estimated by volumetric fusion (Section 5.3) yields only values at discrete voxel locations, we use bilinear interpolation for calculating $d(\mathbf{p})$ and $w(\mathbf{p})$ at intermediate points.

We formulate our objective as minimizing the distance between the TSDF values of the initial reconstruction specified by the mappings $d(\mathbf{p})$ and $w(\mathbf{p})$, and a set of jointly optimized 3D shape models. Fig. 5.7 illustrates two of the 3D shape models learned by our approach. While many choices are possible, for simplicity, we consider a linear embedding to model the shape of the objects.

5.5.1 Notation

Before formulating the objective function, we first introduce our notation based on the TSDF representation as follows:

- N: the number of observations (i.e., 3D bounding box proposals from Section 5.4).

- M: the number of shape models we want to learn.

- $\Omega \subset [0,1]^3$: the discrete domain of our shape models.
  We simply take Ω as an axis aligned 3D grid with equidistant spacing between points, i.e. a unit cube. A 3D shape is represented as the 0-level set of signed distance values $d(\mathbf{p})$ specified at each 3D point $\mathbf{p} \in \Omega$.

- $\pi = \{\pi_1, \ldots, \pi_N\}$: a set of affine mappings from the unit cube to $\mathbb{R}^3$, i.e., $\pi_i : [0,1]^3 \to \mathbb{R}^3$.
  These mappings specify the location of an observed object i in terms of its 3D bounding box, obtained by mapping the edges of the unit cube $[0,1]^3$ via π_i to coordinate system of the initial reconstruction. In this work, we focus our attention on a subset of affine mappings: We assume that all objects are located on a common (and known) ground plane, and thus parametrize each mapping π_i in terms of the following 5 parameters:
  - translation in the 2D ground plane: $\mathbf{t}_i \in \mathbb{R}^2$
  - rotation around the vertical axis: $r_i \in [0, 2\pi]$
  - scaling in all three dimensions: $\mathbf{s}_i \in \mathbb{R}^3_{\geq 0}$

- $\phi = \{\phi_1, \ldots, \phi_M\}$: a set of D-dimensional linear embeddings to model the shape of objects where $D \ll |\Omega|$.
  Each $\phi_j \in \phi$ specifies a real value for every $\mathbf{p} \in \Omega$ given a coefficient vector $\mathbf{x} \in \mathbb{R}^D$, i.e. $\phi_j : \Omega \times \mathbb{R}^D \to \mathbb{R}$. We model the object shape as

$$\phi_j(\mathbf{p}, \mathbf{x}) = \mu_j(\mathbf{p}) + \sum_{d=1}^{D} x_d \, \xi_d^{(j)}(\mathbf{p}) \tag{5.20}$$

  where the parameters $\mu \in \mathbb{R}^{|\Omega|}$ and $\xi_d \in \mathbb{R}^{|\Omega|}$ specify the mean and an orthonormal basis, respectively. Each ϕ_j represents the shape (or TSDF) of model j by specifying a signed distance value at each point $\mathbf{p} \in \Omega$, given an observation dependent coefficient vector $\mathbf{x}$ as input. We specify one coefficient vector for each observation $\mathbf{x}_i \in \mathbb{R}^D$, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. The complexity of the model varies with the dimensionality D. For the simplest case ($D = 0$), we obtain the mean model.

- $\mathbf{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_M\}$: the corresponding set of average scales for M shape models.
  Each shape model ϕ_j is attributed an average scale $\mathbf{v}_j \in \mathbb{R}^3_{\geq 0}$ to encourage scale consistency between different observations associated to that model.

- $\mathbf{k} = \{k_1, \ldots, k_N\}$: the index set which determines the assignment of observations to models, i.e. $k_i \in \{1, \ldots, M\}$.

5.5.2 Objective Function

We formulate our objective as minimizing the distance between the TSDF values of all N observations and the joint model with M shape models. The joint model is specified in terms of the alignment parameters π, shape parameters $(\phi, \mathbf{X}, \mathbf{V})$ and the assignments $\mathbf{k}$. At the same time, we enforce an agreement in scale between different observations associated to a model and regularize against the initial detections. This formulation leads to an objective function defined in terms of a discrete-continuous optimization problem

$$\underset{\pi,\phi,\mathbf{X},\mathbf{V},\mathbf{k}}{\operatorname{argmin}} \sum_{i=1}^{N} \Psi_i(\pi, \phi, \mathbf{X}, \mathbf{V}, \mathbf{k}) \tag{5.21}$$

with energy function

$$\Psi_i(\cdot) = \psi_{shp}(\pi_i, \phi_{k_i}, \mathbf{x}_i) + \lambda_{scale}\, \psi_{scale}(\mathbf{s}_i, \mathbf{v}_{k_i}) + \lambda_{reg}\, \psi_{reg}^{(i)}(\pi_i) \tag{5.22}$$

$\psi_{shp}(\cdot)$ ensures that the shape of observation i fits the associated model k_i, $\psi_{scale}(\cdot)$ encourages agreement in scale, and $\psi_{reg}(\cdot)$ is a regularizer which penalizes strong deviations from the initial detections. $\lambda_{scale}, \lambda_{reg} \in \mathbb{R}_{\geq 0}$ are parameters controlling the influence of the corresponding terms. Next, we define each term in the energy function.

Shape Term: We define the shape term $\psi_{shp}(\cdot)$ as the *weighted squared TSDF difference* between the associated model ϕ and the observation specified via π:

$$\psi_{shp}(\pi, \phi, \mathbf{x}) = \sum_{\mathbf{p} \in \Omega} w(\pi(\mathbf{p}))\, [\phi(\mathbf{p}, \mathbf{x}) - d(\pi(\mathbf{p}))]^2 \tag{5.23}$$

Scale Term: Scale discrepancy is measured by the squared distance between the observation scale $\mathbf{s}$ and the model scale $\mathbf{v}$:

$$\psi_{scale}(\mathbf{s}, \mathbf{v}) = \|\mathbf{s} - \mathbf{v}\|^2 \tag{5.24}$$

Regularizer: Finally, strong deviations from the initial detection i are penalized as

$$\psi_{reg}^{(i)}(\pi_i) = (r_i - \hat{r}_i)^2 + \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|^2 + \|\mathbf{s}_i - \hat{\mathbf{s}}_i\|^2 \tag{5.25}$$

where $(r_i, \mathbf{t}_i, \mathbf{s}_i)$ are the transformation parameters of object i and $(\hat{r}_i, \hat{\mathbf{t}}_i, \hat{\mathbf{s}}_i)$ denote the initial transformation parameters specified by the detection.

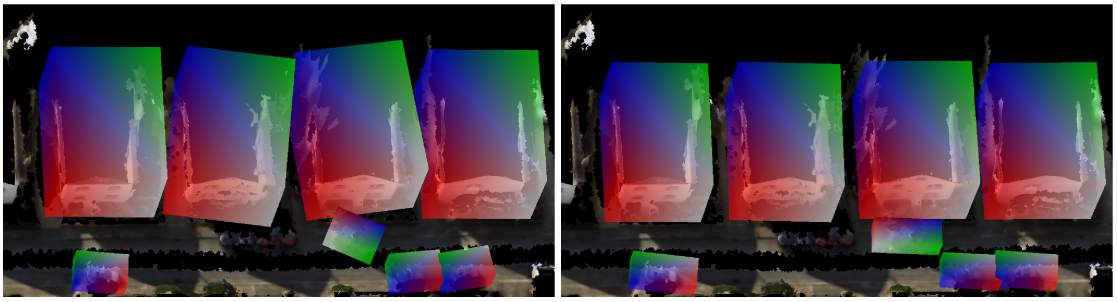


Figure 5.8: **Object Poses.** This figure shows the initial 3D detections (left) and the result after joint alignment and model learning (right) overlayed with the initial reconstruction.

5.6 Inference

Optimizing Eq. 5.21 directly is a very difficult task due to the large dimensionality of the parameter space. We therefore partition the set of variables into tractable blocks and apply block coordinate descent (BCD) to find a local minimizer. Each block lowers the value of the objective function, thus our algorithm is guaranteed to converge. More specifically, we initialize π, $\mathbf{V}$ and $\mathbf{k}$ according to the detections, the mean of ϕ to a random observation, $\mathbf{X} = \mathbf{0}$, and iterate the following blocks:

Block $\{\pi, \mathbf{V}\}$

The first block optimizes the object poses π jointly with the model scales $\mathbf{V}$ while keeping the other variables fixed. Due to the small number of parameters involved, we leverage gradient descent in order to find a local minimum. We first differentiate the objective function in Eq. 5.21 with respect to $\{\mathbf{t}_i, r_i, \mathbf{s}_i\}$ and $\mathbf{V}$ (see Section A.1.1), and then solve the non-linear least squares problem using the Ceres solver Agarwal *et al.* (2010). Fig. 5.8 illustrates the object poses before (left) and after (right) convergence.

Block $\{\phi, \mathbf{X}\}$

The second block optimizes the shape models ϕ jointly with the coefficients $\mathbf{X}$ while keeping the other variables fixed. Note that this optimization does not depend on the scale term $\psi_{scale}(\cdot)$. Further, the object poses π are fixed. Thus the objective in Eq. 5.21 reduces to M independent weighted PCA problems which we solve using the robust approach of Torre et al. la Torre and Black (2001). For the case $D = 0$, this becomes equivalent to computing the weighted mean.

Block $\{\mathbf{X}, \mathbf{k}\}$

The final block optimizes the coefficients $\mathbf{X}$ jointly with the model associations $\mathbf{k}$ while keeping the other variables fixed. We first note that this optimization can be performed

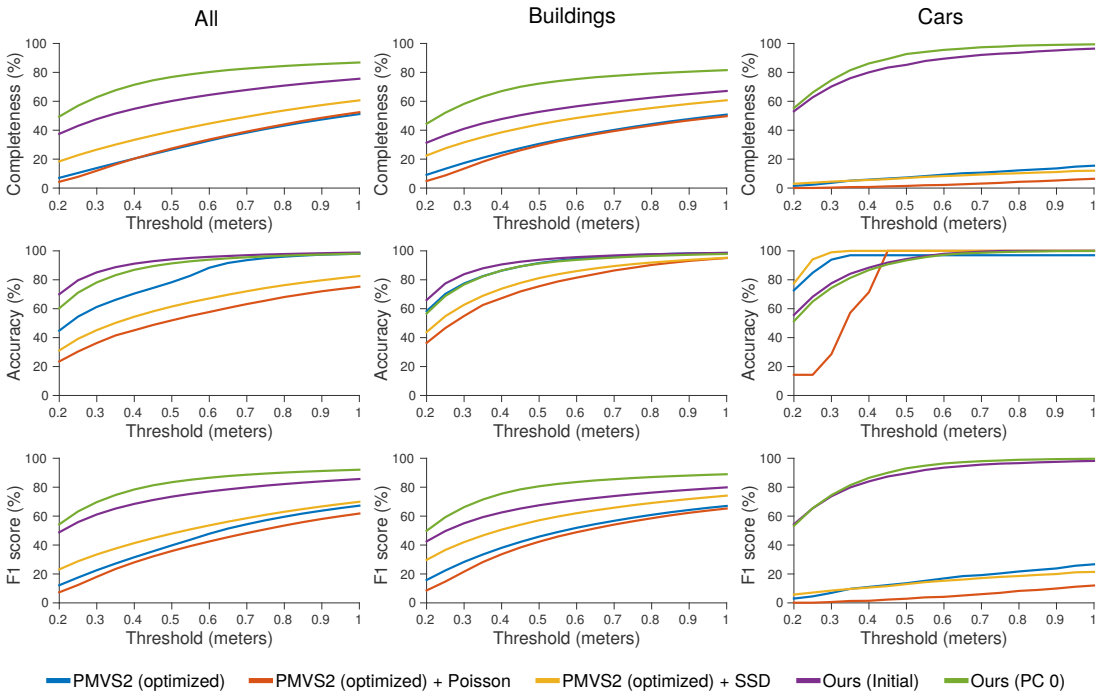


Figure 5.9: **Varying the Evaluation Distance.** This figure shows quantitative results in terms of completeness, accuracy and F1 score when varying the evaluation distance τ between 0.2 and 1.0 m. To avoid clutter, we only show results for a subset of the methods.

independently for each observation $i \in \{1,\dots,N\}$. We thus obtain $\mathbf{x}_i$ and k_i for observation i as

$$\underset{\mathbf{x}_i,k_i}{\operatorname{argmin}} \; \psi_{shp}(\pi_i,\phi_{k_i},\mathbf{x}_i) + \lambda_{scale}\,\psi_{scale}(\mathbf{s}_i,\mathbf{v}_{k_i}) \tag{5.26}$$

which can be found by minimization with respect to $\mathbf{x}_i$ for each $k_i \in \{1,\dots,M\}$.

As $\psi_{scale}(\cdot)$ does not depend on $\mathbf{x}_i$, the minimizer of $\psi_{shp}(\cdot)$ with respect to $\mathbf{x}_i$ can be identified with the solution to an ordinary linear least squares problem (see Section A.1.2).

5.7 Experiments

This section presents results of the proposed approach in comparison to several baselines both quantitatively and qualitatively. As existing datasets for quantitative evaluation of multi-view reconstruction (e.g., Middlebury Seitz *et al.* (2006)) focus on small scenes of single objects where our algorithm is not applicable, we have recorded a novel suburban dataset for our purpose where objects of similar shape such as buildings and cars occur frequently as explained in Section 5.1. As the sequences have been recorded un-

	All		
	Comp.	Acc.	F1 score
PMVS2 (default)	12.24 %	79.26 %	21.20 %
PMVS2 (optimized)	26.65 %	78.31 %	39.77 %
PMVS2 (default) + Poisson	20.90 %	64.12 %	31.53 %
PMVS2 (optimized) + Poisson	27.38 %	51.93 %	35.85 %
PMVS2 (default) + SSD	25.27 %	54.21 %	34.47
PMVS2 (optimized) + SSD	39.27 %	61.51 %	47.94 %
Ours (Initial)	60.21 %	**94.15 %**	73.45 %
Ours (PC 0)	**76.83 %**	91.35 %	**83.46 %**
Ours (PC 1)	76.65 %	89.53 %	82.59 %

	Buildings		
	Comp.	Acc.	F1 score
PMVS2 (default)	16.35 %	85.70 %	27.46 %
PMVS2 (optimized)	30.57 %	91.59 %	45.84 %
PMVS2 (default) + Poisson	25.49 %	77.05 %	38.30 %
PMVS2 (optimized) + Poisson	29.39 %	75.50 %	42.31 %
PMVS2 (default) + SSD	34.12 %	66.32 %	45.05 %
PMVS2 (optimized) + SSD	44.07 %	81.19 %	57.13 %
Ours (Initial)	52.70 %	**93.84 %**	67.50 %
Ours (PC 0)	**72.24 %**	91.26 %	**80.64 %**
Ours (PC 1)	71.55 %	89.46 %	79.51 %

	Cars		
	Comp.	Acc.	F1 score
PMVS2 (default)	0 %	0 %	0 %
PMVS2 (optimized)	7.31 %	96.97 %	13.59 %
PMVS2 (default) + Poisson	1.58 %	**100.00 %**	3.12 %
PMVS2 (optimized) + Poisson	1.46 %	**100.00 %**	2.88 %
PMVS2 (default) + SSD %	0 %	0 %	0 %
PMVS2 (optimized) + SSD	6.94 %	**100.00 %**	12.98 %
Ours (Initial)	85.20 %	94.35 %	89.54 %
Ours (PC 0)	**92.69 %**	93.46 %	**93.07 %**
Ours (PC 1)	91.53 %	92.90 %	92.21 %

Table 5.1: **Quantitative Evaluation.** This figure shows the performance of the baseline methods (PMVS2 variants and initial reconstruction) and our method (PC 0/1) with respect to completeness, accuracy and F1 score using a 0.5 m detection threshold. We separately evaluate all regions within 20 m from the camera, as well as regions representing the categories “buildings” and “cars”.

der regular daylight conditions, they are (unlike Middlebury Seitz *et al.* (2006)) highly challenging for current reconstruction pipelines. Some of the challenges are highlighted in Fig. 5.1. We evaluate all approaches qualitatively and quantitatively with respect to LIDAR ground-truth on the proposed dataset. The parameters in our model have been empirically determined.

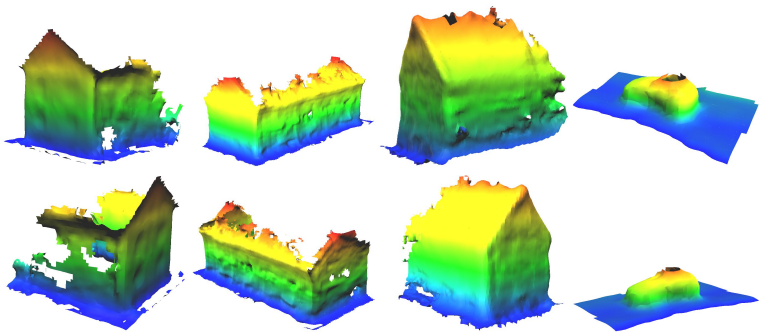
5.7.1 Methods

As baseline, we leverage the state-of-the-art reconstruction pipeline PMVS2 from Furukawa and Ponce (2010) in combination with two meshing alternatives, namely Poisson reconstruction (Kazhdan and Hoppe, 2013) and smooth signed distance surface reconstruction (Calakli and Taubin, 2011). As those surface reconstruction methods tend to produce closed surfaces, we clip large triangles as suggested by Furukawa et al. Furukawa and Ponce (2010). In order to make our data applicable to PMVS2, we projected the fisheye images onto virtual perspective image planes using an opening angle of 120°, such that the images cover all objects which appear in the ground truth. We realized that the performance of PMVS2 heavily depends on the parameter settings. Therefore, we optimized all parameters of the baseline with respect to the reconstruction metrics using grid search on our compute cluster. For completeness, we show PMVS2 results using both the optimized and the default parameters. In addition to the meshed results, we also directly evaluate the point cloud returned by PMVS2. Furthermore, we compare the results of our full model with respect to the initial reconstruction. We leverage the marching cubes algorithm (Lorensen and Cline, 1987) to turn our volumetric reconstruction results into meshes. For all meshes, we remove spurious isolated vertices created by the reconstruction algorithm in a post-processing step. Throughout all experiments, we set the parameters in our model to $\lambda_{size} = 1000$ and $\lambda_{reg} = 50$ which have been determined empirically. Furthermore, as sky regions often lead to spurious matches, we trained a sky detector using the semantic segmentation approach of Ladicky *et al.* (2014a) and removed sky regions before processing the images for all methods.

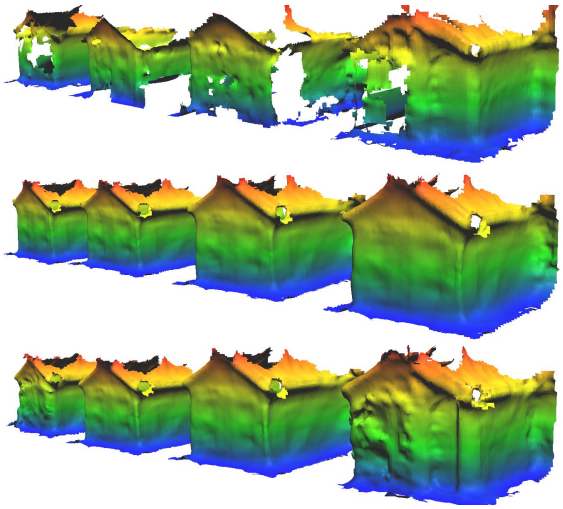
5.7.2 Quantitative Experiments

For quantitative evaluation, we measure performance in terms of completeness, accuracy and F1 score. We calculate completeness as the percentage of ground truth 3D points for which at least one reconstructed 3D point (i.e., vertex of the sub-sampled mesh) is within a distance of $\tau = 0.5$ m. Similarly, we calculate accuracy as the percentage of reconstructed 3D points for which at least one ground truth 3D point is within a distance of $\tau = 0.5$ m. Furthermore, we provide the combined F1 score:

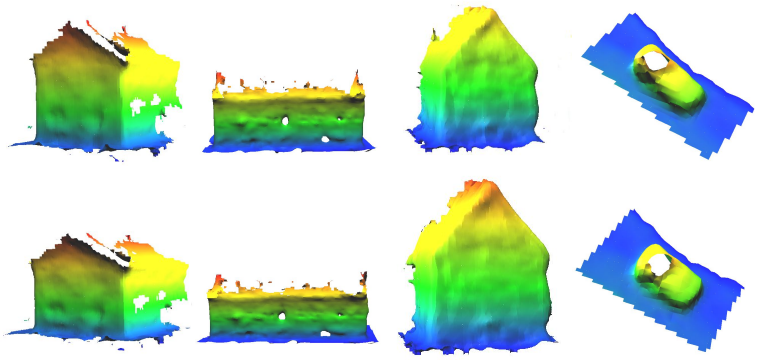
$$F_1 = 2 \cdot \frac{\text{completeness} \cdot \text{accuracy}}{\text{completeness} + \text{accuracy}} \tag{5.27}$$



(a) **Model Variety.**



(b) **Intra-model Variation.**



(c) **Variation in Scale.**

Figure 5.10: **Visualization of Learned Models.** Figure a shows different shape models learned by our approach from two viewpoints. Intra-model shape variations are illustrated in b, where the rows show (top-to-bottom): the input data, the reconstruction with 0 PCs and the reconstruction with 1 PC. Note how the PC 1 model is able to capture the step in the rightmost building while PC 0 enforces stronger regularization. Finally, we show variation in model scale (**s**) after optimization in figure c. Note how the width and height of the objects differs between instances.

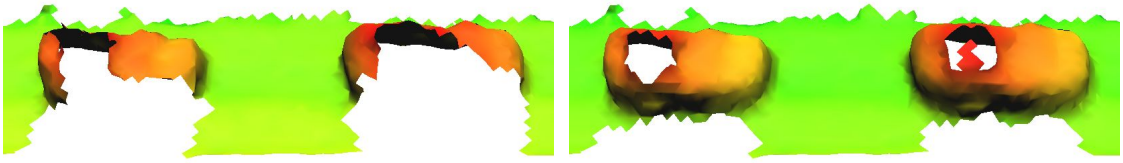


Figure 5.11: **Model Completion.** Our method fills in the occluded/unseen side of objects (left: input; right: our result).

Our quantitative results are shown in Table 5.1, evaluated at all 3D ground truth points (left column), as well as restricted to buildings and vehicles[3] (middle and right column). We evaluate our initial reconstruction as well as our joint reconstruction results for $D = 0$ (PC 0) and $D = 1$ (PC 1). As evidenced by our experiments, our initial reconstruction is able to outperform all variants of PMVS2 (Furukawa and Ponce, 2010) in terms of both completeness and accuracy in almost all regions. Note that for cars, PMVS2 recovers less than 10% of the surfaces due to the challenges in matching textureless and specular surfaces. In contrast, our joint reconstruction (PC 0 / PC 1) transfers surface information from similar shapes in the scene, boosting completeness by more than 15% with respect to to our initial reconstruction baseline with a moderate loss in accuracy. This leads to significant improvements in F1 score for all three categories.

It may seem surprising that our weighted mean model (PC 0) slightly outperforms the more expressive model comprising one principal component (PC 1). After inspection, we found this effect to be caused by systematic errors in the initial reconstruction. This systematic noise can be partially overcome by our mean model (PC 0), which poses a stronger regularization on our joint reconstruction. An alternative for alleviating this effect would be to integrate class-specific object knowledge into the process (Prisacariu and Reid, 2011a,b).

Results for a subset of the methods when varying the evaluation distance threshold τ are illustrated in Fig. 5.9. While completeness and accuracy decreases for all methods with smaller detection thresholds, the relative gain in performance of our method with respect to the baselines increases. This indicates that our reconstructions are not only more complete, but also *metrically* more accurate.

5.7.3 Qualitative Experiments

Our qualitative results are shown in Fig. 5.10-Fig. 5.14. Fig. 5.10 demonstrates the variability of our learned models and Fig. 5.11 illustrates the ability of our model to complete regions of objects which have never been observed. Fig. 5.13 shows (from-top-to-bottom): the point cloud created by PMVS2 (Furukawa and Ponce, 2010) using

[3] In order to evaluate buildings and vehicles separately, we have annotated them with ground truth 3D bounding boxes.

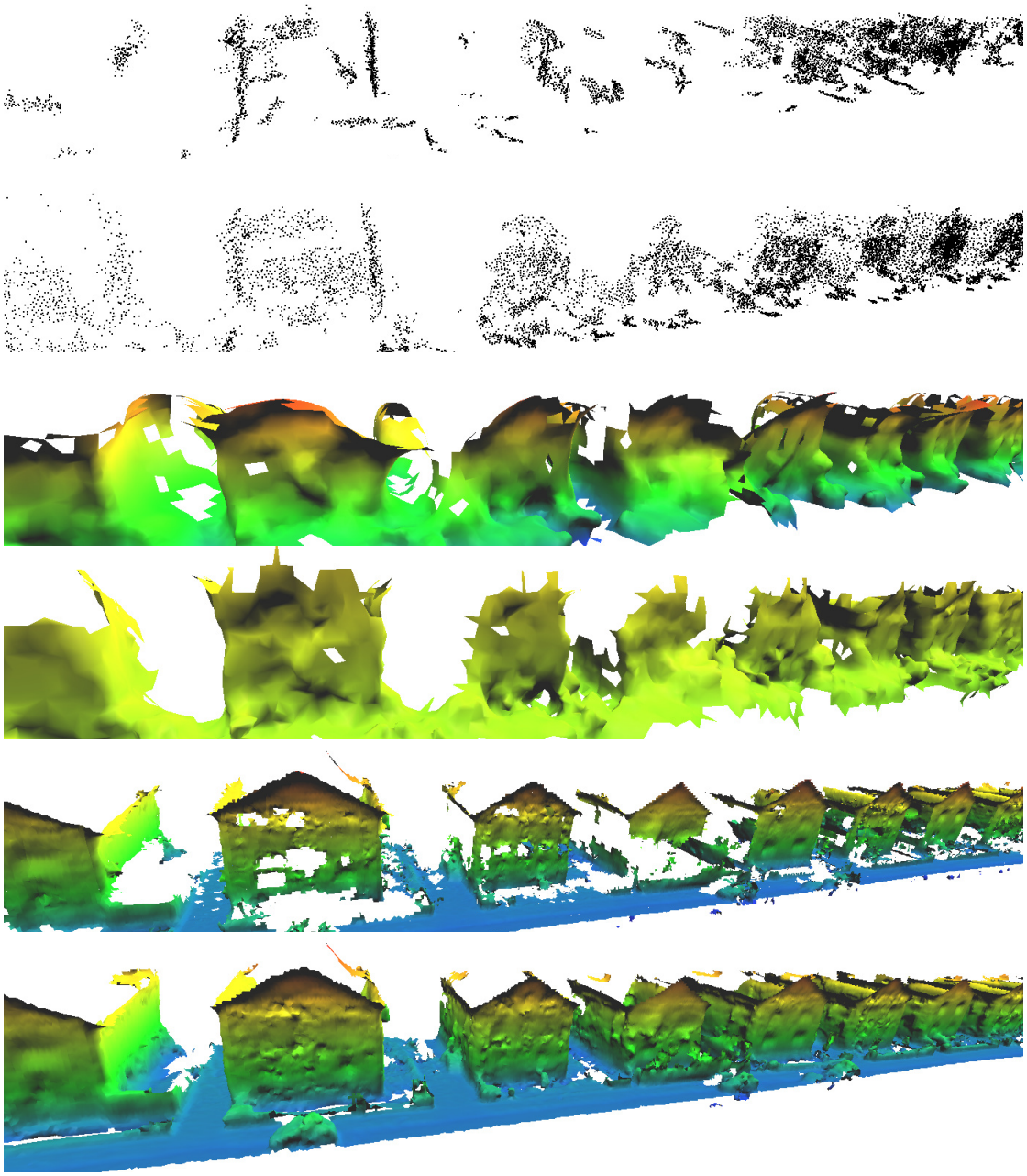


Figure 5.12: **Qualitative Results.** This figure shows our reconstruction results from both sides of the same street, respectively. From top-to-bottom: Point cloud from PMVS2 using the default parameter setting, point cloud from PMVS2 with optimized parameters, PMVS2 (optimized) + Poisson, PMVS2 (optimized) + SSD, our initial reconstruction, and our fused result (PC 0). Note how our method is able to recover cars as well as missing building walls, and handling different types of buildings.

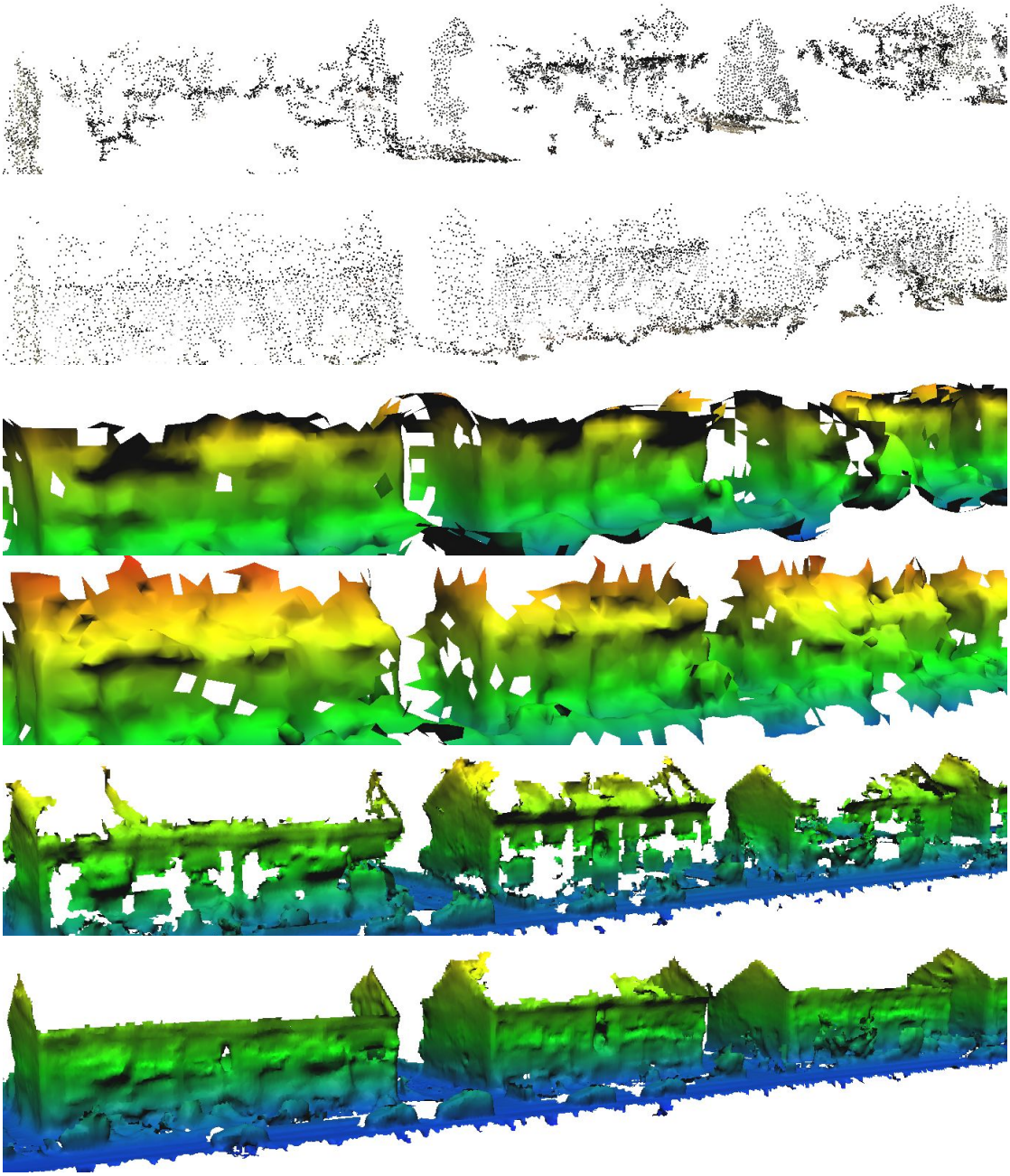


Figure 5.13: **Qualitative Results.** This figure shows our reconstruction results from both sides of the same street, respectively. From top-to-bottom: Point cloud from PMVS2 using the default parameter setting, point cloud from PMVS2 with optimized parameters, PMVS2 (optimized) + Poisson, PMVS2 (optimized) + SSD, our initial reconstruction, and our fused result (PC 0). Note how our method is able to recover cars as well as missing building walls, and handling different types of buildings.

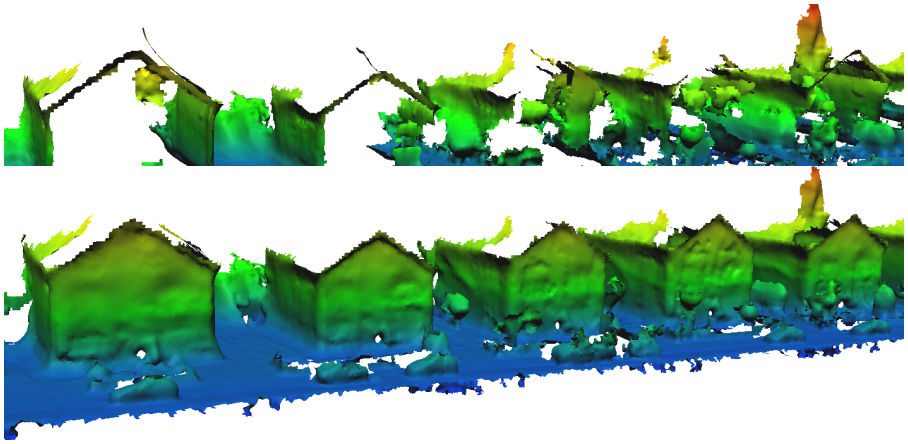


Figure 5.14: **Qualitative Results.** This figure shows the initial reconstruction (top) compared to our joint reconstruction (bottom). Note how the proposed approach can complete the missing walls of buildings even in very challenging cases where there is little surface information in the initial reconstruction.

the default parameter setting, the result with optimized parameters, meshed results of the optimized PMVS2 point cloud using Poisson (Kazhdan and Hoppe, 2013) and SSD (Calakli and Taubin, 2011) surface reconstruction, our initial reconstruction, and our final results (PC 0). The colors denote the height, normalized with respect to the highest and the lowest point of the reconstruction. Note how our method is able to recover cars as well as missing walls of the buildings. Furthermore, our reconstructions convey much more detail than the baseline methods. As roofs in this sequence are barely observed from the viewpoint of the vehicle, none of the algorithms was able to reconstruct them.

We visualize the accuracy (Fig. 5.15) and the completeness (Fig. 5.16) of the different methods for the same sequence fixing the threshold to 0.5 meters. In these plots, accurate or complete points are colored green, while the missing or falsely reconstructed points are colored red. PMVS2 produces accurate but very little number of points, resulting in many red points in the completeness plot. Optimized parameter setting is better compared to the default setting, however, none of the two meshing algorithms we tried can build realistic looking surfaces from the limited set of points produced. Our initial reconstruction is already more complete compared to these results. Joint reconstruction by learning shape of similar objects boosts completeness further, resulting in surfaces that are both complete and accurate. These plots agree with the quantitative results we report in Table 5.1. We also include a zoomed in comparison of our fused result (PC 0) to our initial reconstruction in terms of both accuracy (Fig. 5.17a) and completeness (Fig. 5.17b). Finally, we show additional qualitative results on other sequences of our dataset. Fig. 5.14 shows the benefit of shape modeling for buildings in extreme cases where there is almost no surface reconstructed for some instances after the volumetric fusion.

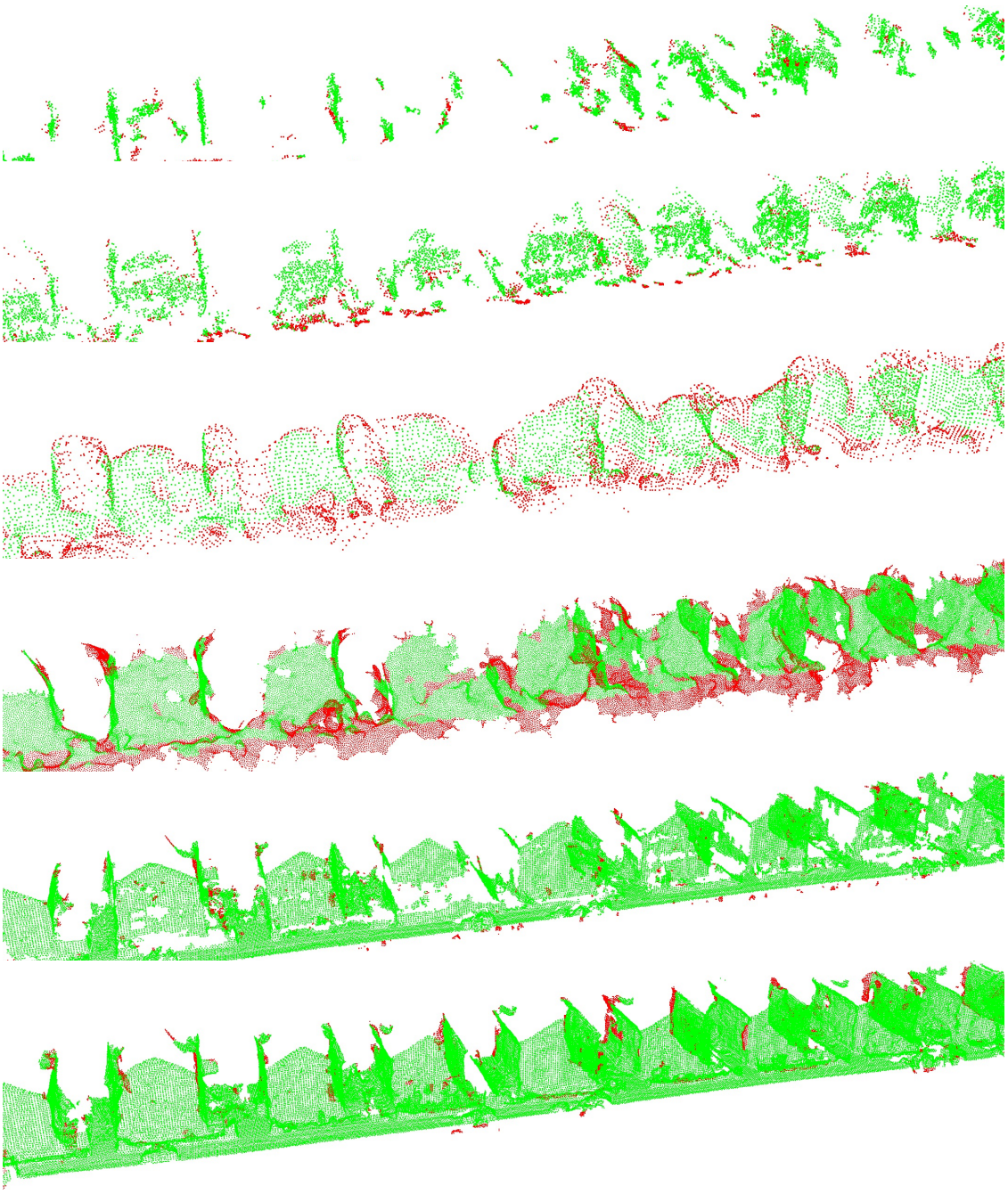


Figure 5.15: **Accuracy Visualization.** From top to bottom: Point cloud from PMVS2 using the default parameter setting, point cloud from PMVS2 with optimized parameters, PMVS2 (optimized) + Poisson, PMVS2 (optimized) + SSD, our initial reconstruction, and our fused result (PC 0). Green-coloured points are reconstructed 3D points for which at least one ground truth 3D point is within a distance of $\tau = 0.5$ m and red-coloured points are other (falsely) reconstructed points.

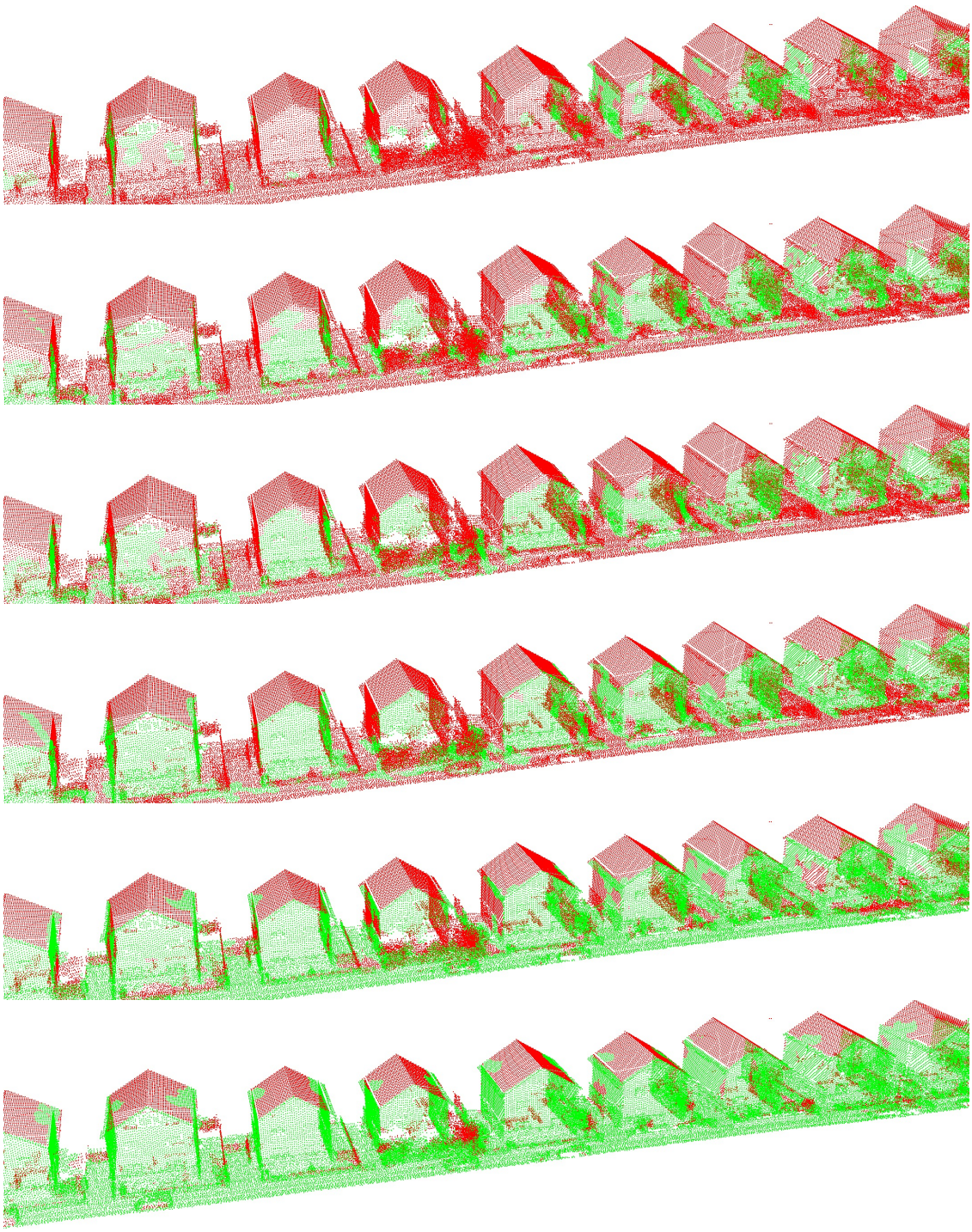
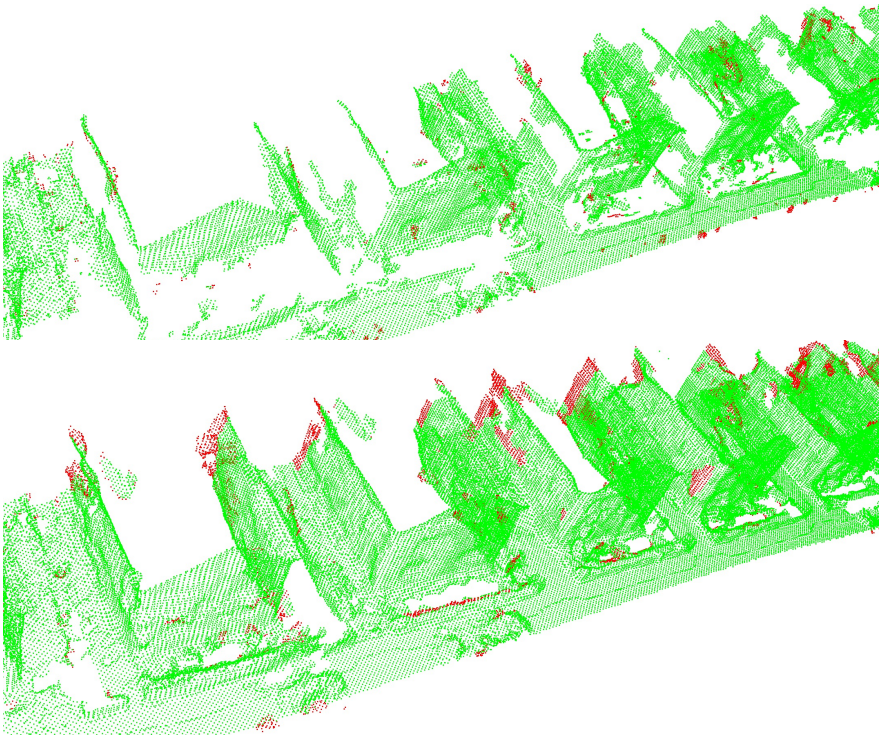
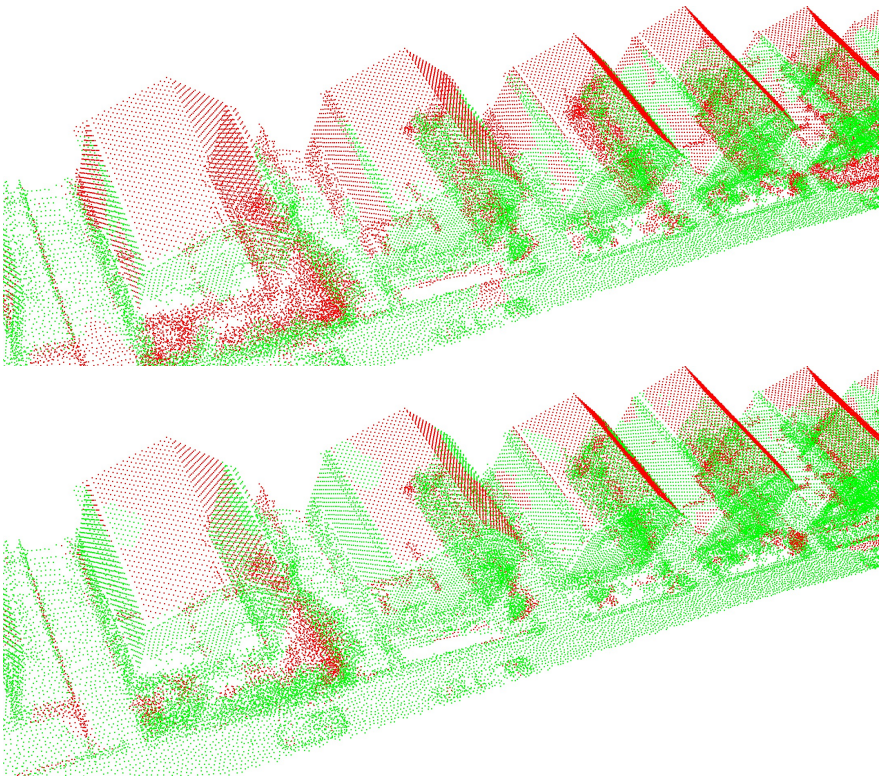


Figure 5.16: **Completeness Visualization.** From top to bottom: Point cloud from PMVS2 using the default parameter setting, point cloud from PMVS2 with optimized parameters, PMVS2 (optimized) + Poisson, PMVS2 (optimized) + SSD, our initial reconstruction, and our fused result (PC 0). Green-coloured points are ground truth 3D points for which at least one reconstructed 3D point is within a distance of $\tau = 0.5$ m and red-coloured points are other (missed) ground truth points.



(a) **Accuracy Visualization.**



(b) **Completeness Visualization.**

Figure 5.17: **Qualitative Results** This figure shows zoomed in comparisons of our initial reconstruction and our fused result (PC 0) in terms of accuracy (a) and completeness (b).

Chapter 6

Learning Features for Optical Flow Estimation

In this chapter, we investigate the utility of feature learning for discrete optical flow approach proposed by Menze *et al.* (2015). In the original Discrete Flow approach of Menze *et al.* (2015), first, a set of proposals are created on the target image for each pixel on the reference image by performing approximate neraest neighbor (ANN) search based on appearance feature vectors. Given these proposals, the optical flow problem is formulated as a discrete MAP inference in a Markov Random Field with pairwise smoothness priors, followed by sub-pixel interpolation (Revaud *et al.*, 2015). We follow their framework, but replace their hand-crafted descriptors with learned feature representations in order to investigate the effect of feature learning on this framework as illustrated in Fig. 6.1. Further, we perform exact matching instead of ANN search by exploiting the compute capability of GPUs and show its advantage over ANN in terms of performance and runtime in our experiments.

Optical flow estimation falls into the category of dense prediction tasks where the goal is to produce an estimation for each pixel in the image. In other words, the prediction is expected to have the same size and structure as the input with accurate estimates at every pixel by respecting the edges. Most of the successful examples of dense prediction tasks such as semantic segmentation require integrating knowledge from a global or multi-scale context and optical flow is no exception. The typical way of achieving this in convolutional neural networks is the repeated combination of max-pooling and convolution with strides at consecutive layers. However, these operations significantly reduce the spatial resolution of the output by violating the full resolution expectation of dense prediction task.

Various solutions have been proposed to handle the requirement of context aggregation while still keeping the full-resolution, typically applied to semantic segmentation. One approach is using multiple rescaled versions of the image as input and combining predictions from each scale in the end. Reasoning over multiple scales is shown to improve the performance for semantic segmentation, but requires processing of feature responses at all layers for multiple scales of the input. Another is adding deconvolutional layers to recover resolution by up-sampling low resolution feature maps. However, this introduces

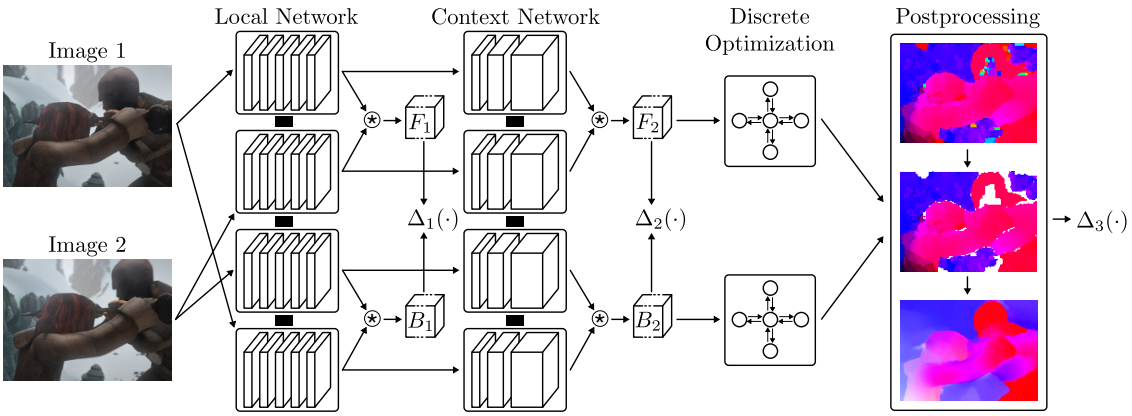


Figure 6.1: **Discrete Flow.** The input images are processed in forward order (top stream) and backward order (bottom stream) using local and context Siamese convolutional neural networks, yielding per-pixel descriptors. We then match points on a regular grid in the reference image to every pixel in the other image, yielding a large tensor of forward matching costs (F_1/F_2) and backward matching costs (B_1/B_2). Matching costs are smoothed using discrete MAP inference in a pairwise Markov random field. Finally, a forward-backward consistency check removes outliers and sub-pixel accuracy is attained using the EpicFlow interpolator (Revaud *et al.*, 2015). We train the model in a piecewise fashion via the loss functions.

additional parameters to learn and requires additional memory and time. Instead, we explore the dilated convolution operation for enlarging the receptive field size of Siamese networks, which is also known as the atrous convolution in the signal processing community (Yu and Koltun, 2016; Chen *et al.*, 2015a). Dilated convolution operations allow to increase the input patch size without increasing the number of parameters or amount of computation compared to a regular convolution.

In particular, we investigate two types of networks: a local network with a small receptive field consisting of 3×3 convolutions followed by non-linearities as well as a subsequent context network that aggregates information over larger image regions using dilated convolutions. As naïve *patch-based* training with dilated convolutions is computationally expensive, we propose an efficient implementation based on regular strided convolutions. For efficient learning of the whole pipeline, we specify auxiliary loss functions akin to Žbontar and LeCun (2016) and train the model in a piecewise fashion.

In Section 6.1, we first introduce Siamese networks for learning similarity measures between image regions (Section 6.1.1) and dilated convolution operation as a way of increasing receptive field size of these networks (Section 6.1.2). Next, we describe our local and context network architecture and provide details of the training and inference procedures. For completeness, we review the discrete optimization framework of Menze *et al.* (2015) and explain the details of post-processing step in Section 6.2. We provide a detailed empirical analysis of the impact of each of the components of the pipeline

in Section 6.3. More specifically, we compare a large number of different local and context architectures with respect to each other and to traditional hand-crafted features. We compare the results of the best-performing systems after discrete optimization and sub-pixel interpolation, and qualitatively visualize the results with their corresponding error images at every stage. Further, we provide a detailed runtime analysis of different design choices and steps of the algorithm.

6.1 Feature Learning using Dilated Convolutions

Following recent trends in computer vision (Han *et al.*, 2015; Simo-Serra *et al.*, 2015; Žbontar and LeCun, 2016; Luo *et al.*, 2016a; Zagoruyko and Komodakis, 2015; Gadot and Wolf, 2016; Bai and Urtasun, 2016), we use deep convolutional neural networks in order to learn better representations tailored for the task. In particular, we use Siamese architectures to process a pair of patches and produce a matching score as an indication of their level of similarity. In addition to traditional local 3×3 convolutional layers, we integrate context information by adopting dilated convolutions. Compared to alternative approaches, dilated convolutions have the advantage of not decreasing the image resolution, thus allowing for efficient dense inference with reuse of computation as explained in Section 6.1.2. In addition, patch-based dilated convolution networks can be efficiently trained as we demonstrate in Section 6.1.4.

For efficiency and due to the difficulty of training CNN-CRF models jointly, we train our model in a piece-wise fashion using auxiliary loss functions. That is, as illustrated in Fig. 6.1, we first train the local architecture using Δ_1, followed by the context architecture using Δ_2, and finally the CRF as well as hyperparameters of the post-processing stage using Δ_3. We also tried joint training on top of the pre-trained local and context networks, but observed no significant improvements. This agrees with the observations reported in Yu and Koltun (2016).

6.1.1 Siamese Networks

Siamese networks was first proposed in the early 1990s by Bromley *et al.* (1993) for solving verification problems. Verification problems in vision are typically based on comparing two images to verify some information such as identity of a person through the signature or face image. In this kind of problem setting, two distinct input points need to be processed to make a decision based on both. A Siamese network consists of two identical branches to process two images that need to be verified. In the end, the outputs of two branches are joined for producing a metric or a score through a loss function defined over both. Two branches are tied through weight sharing during training. Weight sharing between two branches ensures that similar input points are mapped to close locations in the learned feature space. Furthermore, this way, the learned measure

is symmetric, i.e. even the order of the input points changes, the computed metric remains the same.

Matching problem in the context of correspondence search can be considered as a verification problem where the goal is to verify if the two points are the same across different viewpoints or time frames. Siamese network architecture naturally fits to this problem definition by providing a trainable framework to learn nonlinear mappings between input points or more specifically, image regions around the points. Each branch comprises a number convolutional layers with non-linearities in between, typically a rectified linear unit. The output of each branch can be considered as a feature vector which is normalized to unit length when the score is simply defined as the cosine similarity:

$$s(\mathbf{f}_i, \mathbf{f}_j) = \frac{\mathbf{f}_i \cdot \mathbf{f}_j}{\|\mathbf{f}_i\|_2 \, \|\mathbf{f}_j\|_2} \tag{6.1}$$

where s is the score between two feature vectors $\mathbf{f}_i$ and $\mathbf{f}_j$. Normalization reduces the score computation to a simple dot product between all possible pairs. Alternatively, feature vectors can be concatanated without normalization for further processing using a set of fully connected layers followed by the non-linearity units. Žbontar and LeCun (2016) produce the score by using a sigmoid to transform the output of last fully-connected layer in the end. Additional layers for learning the score gives better results for stereo matching, but at a higher cost in terms runtime as explained later.

The goal is to learn discriminative feature representations by mapping corresponding input points closer and non-corresponding ones further away in the feature space. In other words, the similarity metric is maximized for corresponding points and minimized for the others. There are various loss functions designed for that purpose. A typical one is the hinge loss defined over triplets, i.e. the point, the corresponding point, and a non-corresponding point:

$$max(0, m + s_- - s_+) \tag{6.2}$$

where s_+ is the score of the corresponding pair, the s_- is the score of non-corresponding pair and $m \in \mathbb{R}^+$ the margin parameter. This loss is also called margin loss since it tries to create a margin between the score of the corresponding and non-corresponding pairs by maximizing the score for the first and minimizing for the second. More specifically, the loss is zero only when the score of matching sample is larger than the score of non-matching sample by at least some margin m. Otherwise, there is always a loss equal to the difference between $(s_- + m)$ and s_+. The learned scores can be transformed into distances to be used in an energy minimization framework, for example by using cosine distance instead of cosine similarity, i.e. $d = 1 - s$.

An alternative way is to use concatenated images or patches as input. In that case, the processing by the network needs to be repeated for every possible patch pair under consideration. For example, this would be the maximum disparity in stereo matching and the

full image or the number of possible locations inside a maximum flow region in case of optical flow. Using a Siamese network, the outputs of each branch needs to be computed only once because features do not change for different evaluations of score function. Although training is performed on image patches, in testing, the feature computation and the normalization can be performed for all pixels in a single forward pass of the full image which benefits from the reuse of computations in convolutions. The normalized features can be reused for checking the score for possible combinations. However, in case of additional fully connected layers for score computation, every possible combination needs to be propagated through fully connected layers and this creates a bottleneck in terms of performance.

6.1.2 Dilated Convolutions

We leverage dilated convolutions for aggregating more contextual information by increasing the receptive field size. The receptive field of a unit at any layer can be defined as the block of pixels which has an influence on its activation. In contrast to fully connected networks where the value of each unit depends on the entire input, a unit in convolutional networks only depends on a region of the input (Luo *et al.*, 2016b). Ideally, the receptive field should be large enough to cover the image region relevant to the prediction which is particularly important for dense prediction tasks. In this section, we describe dilated convolution operation as an effective way of enlarging the receptive field size compared to regular convolutions. We start by defining regular 2D convolution in discrete domain and build the description of dilated convolution upon it.

Convolution is performed on two functions that are *input* and the *kernel* or the *filter*, and produces a third function as output, which is referred as *filter response*, *activation* or *feature map* in convolutional neural network terminology. More formally, let $\mathbf{I}$ denote a discrete 2D function, i.e. image or the feature map from the previous layer, and let $\mathbf{K}$ denote the filter of size $k \times k$. Regular 2D discrete convolution operator $*$ populates the output $\mathbf{F}$ as defined in the equation:

$$\mathbf{F}_{i,j} = (\mathbf{I} * \mathbf{K})_{i,j} = \sum_m \sum_n \mathbf{I}_{m,n}\, \mathbf{K}_{(i-m),(j-n)} \tag{6.3}$$

The dilated convolution $*_l$ with a dilation factor l is defined as follows:

$$\mathbf{F}_{i,j} = (\mathbf{I} *_l \mathbf{K})_{i,j} = \sum_m \sum_n \mathbf{I}_{m,n}\, \mathbf{K}_{(i-l*m),(j-l*n)} \tag{6.4}$$

In contrast to regular convolutions, the dilated convolution operation reads the input feature map at every l'th location. Note that regular convolution is equal to performing dilated convolution with dilation factor 1, e.g. $l = 1$. In wavelet literature, the dilated convolution is also known as the *hole* algorithm, since it can be considered as up-sampling the filter by the dilation factor and inserting zeroes, e.g. holes, in between the entries of

the filter. The dilated convolution is different than convolution with strides. In case of strided convolution, the filter is applied by shifting at every step while the elements of the filter still remain at adjacent locations on the input. Yu and Koltun (2016) differentiate between dilated convolution and convolution with a dilated filter to emphasize that no dilated filter is constructed, instead the convolution operator is modified to apply the same filter at different ranges using different dilation factors. We follow the same definition and implement it within the torch framework by sparsely sampling the input feature maps.

Receptive Field Size: A common practice in convolutional neural networks is to use spatially small convolutional kernels, typically 3×3. This provides computational efficiency by keeping the number of parameters small while increasing the number of layers. However, in such a convolutional neural network composed of $k \times k$ convolutions as defined above in Eq. 6.3, the receptive field can only grow linearly with the number of layers, i.e. $i(k-1)+k$, where i is the layer index. With a dilated convolution factor l, the receptive field of a filter with the kernel size $k \times k$ becomes $k+(k-1)(l-1)$. Although the effective receptive field increases, the number of filter parameters and the number of operations remain constant.

The effective filter size can be adjusted to expand exponentially with exponentially increasing dilation factors at each layer as in Yu and Koltun (2016). More specifically, for 3×3 kernels and a dilation factor 2^i at layer $i \in \{0,1,...,n-2\}$, the receptive field of a unit becomes $(2^{i+2}-1) \times (2^{i+2}-1)$.

6.1.3 Network Architecture

We use a Siamese network architecture composed of two shared-weight branches, one for the reference patch and one for the target patch. As we are also interested in calculating the backward flow, we have an additional backward Siamese network which shares weights with the forward network as illustrated in Fig. 6.1. Each of the branches consists of several building blocks where each block is defined as convolution, Batch Normalization, and ReLU for non-linearity except the last one which contains only a convolutional layer. The unit-length normalized output of the last layer is used as a feature vector of the patch. The similarity s between image pixels is calculated as the dot product between the respective feature vectors. As opposed to current trends in feature learning for stereo matching (Žbontar and LeCun, 2016), we do not exploit fully connected layers for score computation as the large set of potential correspondences renders this computationally intractable (i.e., one network evaluation for each pixel pair).

Local Network: Our local network leverages 3×3 convolution kernels. The hyperparameters of the network are the number of layers and the number of feature maps

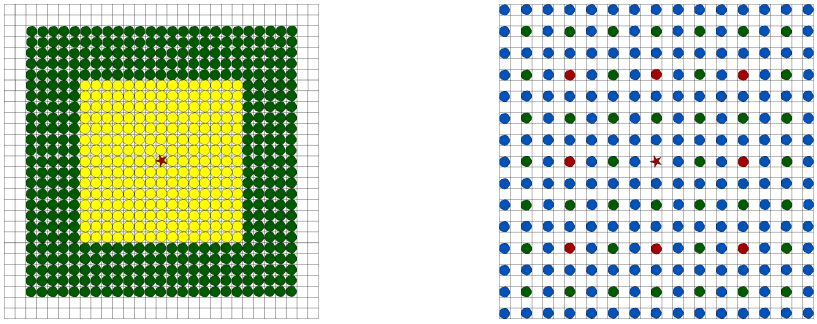
in each layer as specified in our experimental evaluation. We call this network local, because the size of each feature's receptive field is relatively small.

Context Network: Deeper architectures with more convolutional layers increase the receptive field size, possibly leading to improved performance. However, complex high capacity models are also hard to train and require a lot of data. Our context network increases the size of the receptive field with only modest increase in complexity by exploiting dilated convolutions as explained in Section 6.1.2. Dilated convolutions provide more contextual information while not increasing the number of parameters with respect to regular convolutions. Further, in contrast to pooling operations, spatial information is not lost.

6.1.4 Training

We consecutively train the local and the context network using the same auxiliary losses $\Delta_1 = \Delta_2$. As loss function, we leverage the hinge loss function as defined in Eq. 6.2: $\Delta_1(s_-, s_+) = \Delta_2(s_-, s_+) = \max(0, m + s_- - s_+)$. Here s_- denotes the score of a wrong correspondence, s_+ denotes the score of a correct correspondence and m is the margin parameter. We extract positive and negative patch pairs around points with valid ground-truth. Each positive is defined by the ground-truth flow with a perturbation of up to 1 pixel for robustness of the resulting feature representation. Unfortunately, the candidate set for the negative is the whole target image except the ground-truth matching point and thus intractable. Following Žbontar and LeCun (2016), we sample negatives in a small circular region around each positive, keeping a minimum distance from the ground truth location. In particular, we use a threshold of 3 pixels for the minimum distance and a threshold of 6 pixels for the maximum distance to the ground truth flow. This ensures that the training set is composed of patches which are non-trivial to separate.

As illustrated in Fig. 6.2, a naïve implementation of dilated convolutions for training with patches would result in unnecessary computations. As we only need to forward/backward propagate information to/from the center of the patch, we can back-trace the source locations through the dilation hierarchy. Thus, we can implement the dilated convolution operation with sub-sampling and strides as a regular convolution as shown in Fig. 6.2. Furthermore, we are able to exploit the fact that dilated convolutions on patches can be expressed as regular convolutions with strides as illustrated by our pseudo-code in Fig. 6.3. Our experiments show that this reduces computation time as state-of-the-art implementations of regular convolutions (using `cudnn`) are significantly faster compared to dilated ones. This makes training with patch-based dilated convolutional networks much faster. At test time, we reuse the computations by dense convolutions over the image domain in traditional manner.



(a) Naïve (b) Proposed

Figure 6.2: **Dilated Convolution Implementations.** This figure shows the dilated convolution centers on a patch for the context network 2 and 12 with dilation factors 2, 4 and 8 as specified in Tab. 6.5. The center of the patch is marked with a red star and each color corresponds to a convolution center for a specific dilation factor, red for 4 dilations (shown in green), green for 2 dilations (shown in blue) and yellow for both. In other words, red dots show the convolution centers (outputs) for the 4-dilated convolutions which read their input values at the green points. Note that yellow points are only visible in (a) as red and green dots do not overlap in (b) due to the sparsity exploited by our approach.

for $i = 1$ *to* $\#dilations$ **do**
  DilatedConvolution with $dilations[i]$
  if $i < \#dilations$ **then**
    Batch Normalization, ReLU

Sub-sampling with $dilations[1]$
for $i = 1$ *to* $\#dilations$ **do**
  if $i == \#dilations$ **then**
    $stride = 1$
  else
    $stride = \frac{dilations[i+1]}{dilations[i]}$
  Convolution with $stride$
  if $i < \#dilations$ **then**
    Batch Normalization, ReLU

Figure 6.3: **Fast Patch-based Training of Dilated Convolutional Networks. Left:** A naïve implementation requires dilated convolution operations which are computationally less efficient than highly optimized `cudnn` convolutions without dilations. **Right:** The behavior of dilated convolutions can be replicated with regular convolutions by first sub-sampling the feature map and then applying 1-dilated convolutions with stride. Here *dilations* denotes the array that specifies the dilation factor of the dilated convolution in each convolutional layer.

6.1.5 Inference

Differently from training with image patches, at test time the outputs of both branches of the network are computed for each point only once in a single forward pass of the full image, thereby reusing computations. The score computation between multiple reference points and every point on the target image can be performed efficiently as a single big matrix multiplication on the GPU. The first matrix is constructed by stacking reference feature descriptors as rows and the second matrix is built by stacking the target feature descriptors as columns. This waives the need for approximate search strategies as required in CPU-only model (Menze *et al.*, 2015). In our implementation, we handle the large GPU memory requirements by dividing the first matrix into individual chunks, balancing memory usage and computation time. We are able to further cut down inference time, as the post-processing stage which we use requires only every fourth pixel to be matched.

6.2 Discrete Optimization and Post-processing

We follow the Discrete Flow approach (Menze *et al.*, 2015) to aggregate information while respecting uncertainty in the matching. More specifically, we select the 300 best feature match hypotheses for each pixel on a regular 4-spaced grid. The set of hypotheses constitute a proposal set on the target image for each pixel on the reference image. In order to enrich the proposal set, we ensure the uniqueness of proposals and increase the variety by using non-maximum-suppression (NMS) constraints with a threshold of 2 pixels, i.e. any two proposal flow vectors are at least 2 pixel away in both horizontal and vertical direction. In addition, we copy proposal flow vectors from randomly sampled neighboring pixels, as spatially close regions usually have similar optical flow. We sample the neighboring pixel by assuming a local Gaussian distribution centered at the reference pixel and add the best neighbor flow which is not already in the proposal set and which do not violate NMS constraints. In our experiments, we optimize the proportion of number of proposals from neighboring pixels over the total number of proposals as well as the standard deviation of the Gaussian distribution.

The proposal set serves as a discrete set of possible flow vectors for each pixel on the reference image. We formulate optical flow estimation as MAP inference in a discrete MRF. More formally, let $\mathbf{f_p}$ denote the discrete flow associated with pixel $\mathbf{p}$ on the reference image. We minimize the following energy with pairwise smoothness constraints:

$$E(\mathbf{f}) = \sum_{\mathbf{p}} \phi_{\mathbf{p}}(\mathbf{f_p}) + \lambda \sum_{\mathbf{q} \in N_{\mathbf{p}}} \psi_{\mathbf{p},\mathbf{q}}(\mathbf{f_p}, \mathbf{f_q}) \tag{6.5}$$

where $N_{\mathbf{p}}$ denotes the 4-neighborhood of pixel $\mathbf{p}$ on the grid and λ is the relative weight of the pairwise term with respect to the data term.

The data term $\phi_{\mathbf{p}}(\mathbf{f_p})$ encodes the cost at pixel $\mathbf{p}$ given the proposal flow $\mathbf{f_p}$ as the

truncated cosine distance between the corresponding descriptors:

$$\phi_{\mathbf{p}}(\mathbf{f}_{\mathbf{p}}) = \min(1 - \langle \mathbf{d}_{\mathbf{p}}, \mathbf{d}'_{\mathbf{p}+\mathbf{f}_{\mathbf{p}}} \rangle, \tau_{\phi}) \tag{6.6}$$

where $\mathbf{d}_{\mathbf{p}}$ denotes the descriptor vector at pixel $\mathbf{p}$ on the reference image, $\mathbf{d}'_{\mathbf{p}+\mathbf{f}_{\mathbf{p}}}$ denotes the descriptor vector at the corresponding location $\mathbf{p}+\mathbf{f}_{\mathbf{p}}$ on the target image and τ_{ϕ} is the truncation threshold. Further, $\langle \cdot, \cdot \rangle$ denotes the the cosine similarity between two feature vectors and the cosine distance which is $1 - \langle \cdot, \cdot \rangle$ is used as the cost function. Since the descriptors from the last layer of the Siamese Network are already normalized to unit-length, cosine similarity between the descriptors becomes an inner product evaluation, i.e. $\langle \mathbf{d}_{\mathbf{p}}, \mathbf{d}_{\mathbf{p}+\mathbf{f}_{\mathbf{p}}} \rangle = \mathbf{d}_{\mathbf{p}} \cdot \mathbf{d}_{\mathbf{p}+\mathbf{f}_{\mathbf{p}}}$.

The second term in Eq. 6.5 encourages smooth flow fields by minimizing truncated $L1$ distance between flow vectors of neighboring pixels:

$$\psi_{\mathbf{p},\mathbf{q}}(\mathbf{f}_{\mathbf{p}}, \mathbf{f}_{\mathbf{q}}) = \min(\|\mathbf{f}_{\mathbf{p}} - \mathbf{f}_{\mathbf{q}}\|_1, \tau_{\psi}) \tag{6.7}$$

where τ_{ψ} denotes the truncation threshold for the pairwise term. The computational complexity of pairwise term due to large label set can be reduced by exploiting the truncated form of the pairwise potentials (Menze *et al.*, 2015). This results from the observation that a large percentage of pairwise evaluations are truncated due to the diversity of proposal sets. In particular, flow proposals of each neighboring pixel pair $(\mathbf{p}, \mathbf{q})$ are checked for truncation prior to the inference and indices and values of non-truncated ones are stored to be reused during inference. This way, number of evaluations required for the pairwise term are reduced from the square of number of proposals to the number of proposals remaining after the truncation which is shown to be much smaller for practical values of τ_{ψ} by Menze *et al.* (2015). The pre-computation of pairwise values can be further accelerated using hash maps for efficiently retrieving flow vectors.

Menze *et al.* (2015) perform inference using Block Coordinate Descent which iteratively updates alternating image rows and columns conditioned on the MAP solution of the remaining variables for computational efficiency. We found out that max-product loopy belief propagation with reduced number of proposals and efficient robust pairwise potentials results in similar performance.

6.2.1 Post-processing

As some pixels are occluded (i.e., not matchable) and due to the occurrence of outliers, we post-process our results using a forward-backward consistency check after discretely optimizing the forward and the backward flow. We further remove unlikely small segments from the solution using connected-component analysis. The resulting semi-dense flow map is fed into Epic Flow (Revaud *et al.*, 2015) for further refinement to sub-pixel accuracy. We optimize the parameters of the MRF and the post-processing stage using block coordinate descent with a $0/1$ outlier loss $\Delta_3 = [\|\mathbf{f} - \hat{\mathbf{f}}\|_2 > 3\,\mathrm{Px}]$, averaged over all

Arch.	Layers	Feature Maps									RFS
1	5	64	64	64	64	64					11
2	7	64	64	64	64	64	64	64			15
3	7	64	64	64	128	128	128	64			15
4	9	64	64	64	64	64	64	64	64	64	19
5	9	32	32	32	64	64	64	128	128	128	19

Table 6.1: **Local Architectures.** RFS denotes the receptive field size in pixels.

non-occluded pixels. Here, $\mathbf{f}$ is the ground truth flow vector, $\hat{\mathbf{f}}$ denotes its prediction and $[\cdot]$ is the Iverson bracket.

6.3 Experiments

We evaluate the performance of different local and context architectures, as well as the whole Deep Discrete Flow pipeline on MPI Sintel (Butler *et al.*, 2012), KITTI 2012 (Geiger *et al.*, 2012) and KITTI 2015 (Menze and Geiger, 2015). Towards this goal, we trained separate networks for Sintel and KITTI, but merged the training sets of KITTI 2012 and KITTI 2015. For our internal evaluations, we follow the KITTI and MPI Sintel protocols: we split the training set into a training and a validation set using every fifth image for validation and the remaining images for training. While the images in MPI Sintel are temporally correlated, we found that Siamese networks without fully-connected layers do not suffer from over-fitting (i.e., the training and the validation errors behave similarly).

Note that the MPI Sintel and KITTI datasets leverage different evaluation metrics, average endpoint error (EPE) and outlier ratio, respectively. We follow each dataset's criteria for the final results, but report 3 pixel outlier ratios in all non-occluded regions for comparing the raw output of different network architectures, since the primary goal of our learned patch representations is to reduce the number of outliers.

Before passing them to the network, we normalize each image to zero mean and unit variance. Following common wisdom (Simonyan and Zisserman, 2015), we set the kernel size to 3 and use stride 1 convolutions unless otherwise specified. We start the training with standard uniform initialization in Torch and monitor the average outlier ratio in non-occluded regions on a subset of the validation set to stop training. We use stochastic gradient descent with momentum 0.9 for optimization, a batch size of 128, a hinge loss margin of 0.2 and a learning rate of 0.002 without any decay. We observe no over-fitting for neither our local nor our context networks.

Arch.	Out-Noc
1	24.61 %
2	20.54 %
3	20.69 %
4	19.34 %
5	18.31 %

(a) **MPI Sintel**

Arch.	Out-Noc
1	34.60 %
2	29.71 %
3	29.89 %
4	30.37 %
5	27.36 %

(b) **KITTI**

	DF	Optimized
MPI Sintel	29.97 %	19.16 %
KITTI	34.29 %	22.59 %

(c) **Daisy**

Table 6.2: **Comparison of Local Architectures**. Fig. (a)+(b) show the performance of different local architectures using winner-takes-all on the validation sets of MPI Sintel and KITTI, respectively. As baseline, Fig. (c) shows the performance of matching Daisy features on both datasets using the parameter setting of Menze *et al.* (2015) in the first column and our re-optimized parameters in the second column. All numbers are percentages of non-occluded bad pixels as defined by the KITTI evaluation protocol.

6.3.1 Baseline for Feature Matching

We leverage the winner-takes-all solution of Daisy (Tola *et al.*, 2010) feature matching as pursued in Discrete Flow (Menze *et al.*, 2015) as local baseline for our learned feature representations. For a fair comparison, we optimized the hyper-parameters of the Daisy feature descriptor (Tola *et al.*, 2010) on a subset of the MPI Sintel training set using block coordinate descent to minimize the ratio of outliers. The results are shown in Table 6.2c. Note that the optimized Daisy descriptor has 264 dimensions and a receptive field of approximately 40×40 pixels while Discrete flow (Menze *et al.*, 2015) uses Daisy descriptors of length 68 with a receptive field of approximately 20×20 pixels. All numbers correspond to the WTA solution calculated over the full target image using exact matching.

6.3.2 Approximate vs. Exact Matching

Formulating optical flow as an approximate nearest neighbour (ANN) search on the target image is a common way of dealing with the size of the search space (Bao *et al.*, 2014; Besse *et al.*, 2014; Menze *et al.*, 2015). In this section, we show that exact matching can be done as efficiently utilizing the GPU and show improved results compared to ANN.

We can perform exact matching for grid points on the reference image efficiently as matrix-matrix multiplication on the GPU. However, there are still some restrictions due to the limited memory which prevent performing the entire computation at once. We show our experiments by dividing the set of grid point into chunks and changing the number of chunks to find a compensation between memory usage and runtime for different number of matches per grid point in Table 6.3.

For comparison, we use the popular FLANN library (Muja and Lowe, 2014) (which has also been used in Menze *et al.* (2015)) to perform approximate nearest neighbor

Number of Chunks	Memory	Run-time
9	6.63	8.42
18	3.89	8.34
45	2.25	3.03
90	1.71	3.58
180	1.43	4.52
306	1.32	6.94

(a) **Number of Matches = 1**

Number of Chunks	Memory	Run-time
9	6.70	5.58
18	3.94	5.74
45	2.29	6.53
90	1.74	6.94
180	1.46	7.36
306	1.35	8.42

(b) **Number of Matches = 1024**

Table 6.3: **Exact Matching.** This table compares different number of chunks used to compute the exact matching in terms of memory and runtime for two different number of matches, 1 as in WTA and 1024 as used in BP.

matching. We found that the FLANN parameter "number of checks" which specifies the maximum number of leafs to visit when searching for neighbors has a clear effect on the performance. Therefore, we vary the number of checks in Table 6.4, fixing the number of search trees to 8. For both datasets, only using 1024 checks yields performance comparable to exact matching. We also note that runtime for FLANN is much higher, 45 to 60 seconds depending on the number of checks, when using full resolution images and a large number of feature maps, 256 as in case of context architecture 12.

6.3.3 Convergence

In Fig. 6.4, we plot the average outlier ratio in non-occluded regions over the training iterations for both local and context networks on the KITTI training and validation sets. We also note that training the context network on top of the local network decreases outlier ratios significantly. Importantly, the training and validation curves behave similarly for both networks, indicating that our Siamese architectures do not suffer from over-fitting. In both plots, one iteration corresponds to one hundred thousand batch updates.

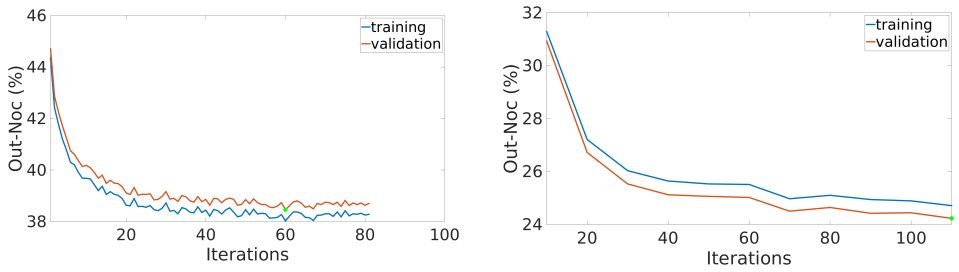
Matching	Number of Checks	Out-Noc
	32	30.25%
FLANN	128	21.21 %
	1024	14.56 %
Exact	-	12.19 %

(a) **MPI Sintel**

Matching	Number of Checks	Out-Noc
	32	56.09%
FLANN	128	41.09 %
	1024	26.19 %
Exact	-	20.28 %

(b) **KITTI**

Table 6.4: **Comparison to FLANN.** This table compares the performance of exact matching to approximate matching using the FLANN library (Muja and Lowe, 2014), varying the number of checks which specify the maximum number of leafs to visit.



(a) **Local Network** (b) **Context Network**

Figure 6.4: **Outlier ratio in non-occluded regions over training iterations.** This figure shows the average outlier ratio in non-occluded regions over the training iterations on KITTI.

6.3.4 Comparison of Local Architectures

We first compare five different local network architectures, including some of the architectures proposed in the literature for feature matching. Our starting point is the simple 5-layer architecture of Žbontar and LeCun (2016) (architecture 1). We create additional architectures by changing the number of layers and feature maps in each layer as shown in Table 6.1. Architecture 5 corresponds to the recently proposed 9 layer network for stereo and flow in Bai and Urtasun (2016); Luo *et al.* (2016a). Adding more layers changes network's receptive field size and has a clear effect on the performance as shown in Table 6.2. However, compared to our local architectures, the Daisy descriptor is fairly competitive. We attribute this to its larger receptive field size. In the next section, we explore context architectures to increase the receptive field size of our learned representations.

6.3.5 Comparison of Context Architectures

Towards this goal, we fix the local architecture to the simplest one (architecture 1), and train different context architectures on top of this network. In a later section, we also show the performance of the best context architecture trained on top of the best local architecture. We create two types of context architectures, one by fixing the number of feature maps to 64, and one by changing the number of feature maps in each layer as summarized in Table 6.5.

Again, we compare the winner-takes-all performance in Table 6.6. We first note that the outlier ratio is significantly lower than the outlier ratio of the local architectures and the Daisy baseline shown in Table 6.2. Secondly, architectures which double the number of feature maps consistently outperform their respective constant counterparts. Finally, the 3-layer context architectures 2 and 12 are the best performing models in our set.

Arch.	Arch.	Feature Maps						Dilations						RFS
1	11	64	128	256	512			2	4	8	16			+60
2	12	64	128	256				2	4	8				+28
3	13	64	128					2	4					+12
4	14	64	128	256				4	8	16				+56
5	15	64	128					8	16					+48
6	16	128	128					4	4					+16
7	17	64	64	128	128			2	2	4	4			+24
8	18	64	64	128	128	256	256	2	2	4	4	8	8	+28
9	19	128	128	256	256			8	8	16	16			+48

Table 6.5: **Context Architectures.** This table shows different context architectures and their receptive field sizes (RFS). We list the architectures that share the same set of dilations (and consequently RFS) in one row. Architectures in the same row differ solely by the number of feature maps in each layer. All the single digit architectures have 64 feature maps at each layer, while the set of feature maps for the double digit ones are shown in the third column. Receptive field sizes are added (+) to the RFS size of the respective local architecture.

Arch.	Out-Noc	Arch.	Out-Noc
1	16.32 %	11	11.92 %
2	14.51 %	12	12.19 %
3	15.65 %	13	14.32 %
4	15.27 %	14	12.53 %
5	15.66 %	15	13.34 %
6	15.10 %	16	13.59 %
7	15.68 %	17	13.69 %
8	15.50 %	18	13.01 %
9	20.04 %	19	14.55 %

(a) **MPI Sintel**

Arch.	Out-Noc	Arch.	Out-Noc
1	30.16 %	11	24.28 %
2	25.82 %	12	20.28 %
3	24.67 %	13	21.39 %
4	29.40 %	14	25.29 %
5	28.54 %	15	24.89 %
6	25.11 %	16	21.13 %
7	26.78 %	17	22.93 %
8	31.63 %	18	24.68 %
9	40.12 %	19	34.43 %

(b) **KITTI**

Table 6.6: **Comparison of Context Architectures.** This table shows the performance of different context architectures on top of local architecture 1 on the validation sets of MPI Sintel and KITTI. "Out-Noc" is defined as in Table 6.2.

6.3.6 Evaluation of Model Components

Table 6.7 compares (from left to right) the results of winner-takes-all (WTA), discrete optimization and the full pipeline including post-processing and Epic Flow interpolation with respect to each other. For this experiment, we selected the simplest local architecture 1 as well as the top performing local architecture 5, both with and without context. For comparison, we also show the results of Discrete Flow with Daisy features as baseline ("DF Menze *et al.* (2015)").

Local	Context	Winner-takes-All				Discrete Optimization				Full Pipeline			
		Noc		Occ		Noc		Occ		Noc		Occ	
		Out	EPE	Out	EPE	Out	EPE	Out	EPE	Out	EPE	Out	EPE
DF Menze *et al.* (2015)		26.36 %	27.92	29.85 %	33.93	10.45 %	7.44	14.82 %	13.05	8.20 %	2.77	11.28 %	4.61
1	-	24.67 %	49.86	28.45 %	62.05	12.06 %	10.27	16.55 %	17.60	7.27 %	2.78	10.14 %	4.41
1	12	12.24 %	25.70	16.76 %	39.69	8.95 %	8.73	13.44 %	16.03	6.93 %	2.61	9.92 %	4.18
5	-	18.36 %	42.51	22.64 %	56.30	10.75 %	10.76	15.29 %	18.61	7.28 %	2.83	10.12 %	4.37
5	12	12.13 %	27.94	16.66 %	42.42	8.75 %	9.12	13.26 %	16.70	7.07 %	2.73	10.02 %	4.29

(a) **MPI Sintel**

Local	Context	Winner-takes-All				Discrete Optimization				Full Pipeline			
		Noc		Occ		Noc		Occ		Noc		Occ	
		Out	EPE	Out	EPE	Out	EPE	Out	EPE	Out	EPE	Out	EPE
DF Menze *et al.* (2015)		33.01 %	30.21	40.99 %	49.16	10.84 %	3.73	21.81 %	22.38	8.55 %	1.76	18.43 %	4.49
1	-	34.38 %	69.45	42.00 %	99.55	13.38 %	8.75	24.00 %	29.92	8.35 %	2.14	16.73 %	4.44
1	12	20.00 %	41.67	29.33 %	77.21	13.26 %	9.90	23.54 %	31.00	9.75 %	2.57	18.27 %	5.35
5	-	27.18 %	58.58	35.69 %	92.92	13.07 %	9.63	23.63 %	31.70	8.74 %	2.38	17.12 %	4.77
5	12	22.09 %	55.46	31.10 %	92.85	14.01 %	12.78	24.16 %	34.72	10.62 %	2.99	19.10 %	5.95

(b) **KITTI**

Table 6.7: **Comparison of Model Components.** This table shows our results after winner-takes-all feature matching, after discrete optimization and the results of the full pipeline including post-processing and sub-pixel interpolation. We report end-point-errors (EPE) and outliers ratios (Out) both in non-occluded (Noc) and in all image regions (Occ) on the respective validation sets.

We first note that for WTA (first column), the context architectures improve the outlier ratio significantly with respect to local architectures for both datasets. However, this improvement is less visible after spatial smoothing (second and third column). We conclude that the gain of leveraging a larger receptive field can be partially compensated by using a spatial smoothing stage.

On the KITTI dataset, the improvements are less pronounced than on the MPI Sintel dataset. Here, our smallest local architecture (second row) outperforms Discrete Flow (Menze *et al.*, 2015) slightly. Interestingly, the context architectures improve performance when considering the winner-takes-all (WTA) solution, but perform on par or even lead to degradation after spatial smoothing (second and third column). Our investigations revealed that the reason for this is the scale changes which are prominently present on KITTI (but less so on MPI Sintel) and which the networks have difficulty to cope with. We thus conclude that progress in invariant deep representations (in particular scale invariance) is necessary to address this issue.

6.3.7 Results on Test Set

We submitted our results to the MPI Sintel and KITTI 2012 and 2015 evaluation servers. We picked the best row for each dataset according to the results in Table 6.7, i.e., local model 1 in combination with context model 12 for the MPI Sintel and local model 1 alone for both KITTI datasets. In accordance with our results on the training/validation

Context Arch.	Naïve	Proposed	Context Arch.	Naïve	Proposed
1	543.37 ms	15.17 ms	11	823.37 ms	42.46 ms
2	146.69 ms	4.80 ms	12	184.88 ms	9.28 ms
3	58.37 ms	2.23 ms	13	69.17 ms	2.79 ms
4	360.41 ms	6.40 ms	14	417.39 ms	11.42 ms
5	195.22 ms	3.60 ms	15	207.55 ms	4.21 ms
6	63.47 ms	2.27 ms	16	95.34 ms	4.41 ms
7	170.60 ms	7.30 ms	17	201.37 ms	9.39 ms
8	646.11 ms	38.77 ms	18	867.846 ms	55.78 ms
9	1384.48 ms	11.00 ms	19	2335.74 ms	26.96 ms

Table 6.8: **Patch-based Dilated Convolutions.** This table compares the runtime of one batch update in milliseconds for the naïve and the proposed implementations using different context architectures with dilated convolutions.

split, we obtain good results on MPI Sintel while we are slightly better than Discrete Flow (Menze *et al.*, 2015) on KITTI 2012 and KITTI 2015. We refer to the benchmark websites for details[1].

6.3.8 Qualitative Results

Fig. 6.5 shows visualizations of the different stages of our approach for several selected images from both MPI Sintel (top) and KITTI (bottom). Some failure cases are shown in Fig. 6.6. Each sub-figure shows from top-to-bottom: the input image and the ground-truth flow, Discrete Flow with Daisy features, our local architecture 1, our architectures 1 + 12. For each sub-figure, the first double column shows the WTA result on the grid, the second the results of discrete optimization and the last double column shows the final result.

As evidenced from these results, the proposed feature learning approach handles object boundaries more precisely and in general leads to lower errors for all inliers. However, these advantages diminish after discrete optimization and in particular Epic Flow interpolation as non-matched regions are responsible for the largest portion of the remaining errors. From Fig. 6.6, it is clearly visible that the learned representations suffer from strong scale changes which need to be addressed to further improve performance.

6.3.9 Runtime Analysis

Runtime of Patch-Based Dilated Convolutional Networks

As mentioned in Section 6.1.4, a naïve implementation of patch based dilated convolutions is computationally very inefficient. In Table 6.8, we show the speed-ups we gain

[1] `http://sintel.is.tue.mpg.de/` `http://www.cvlibs.net/datasets/kitti/`

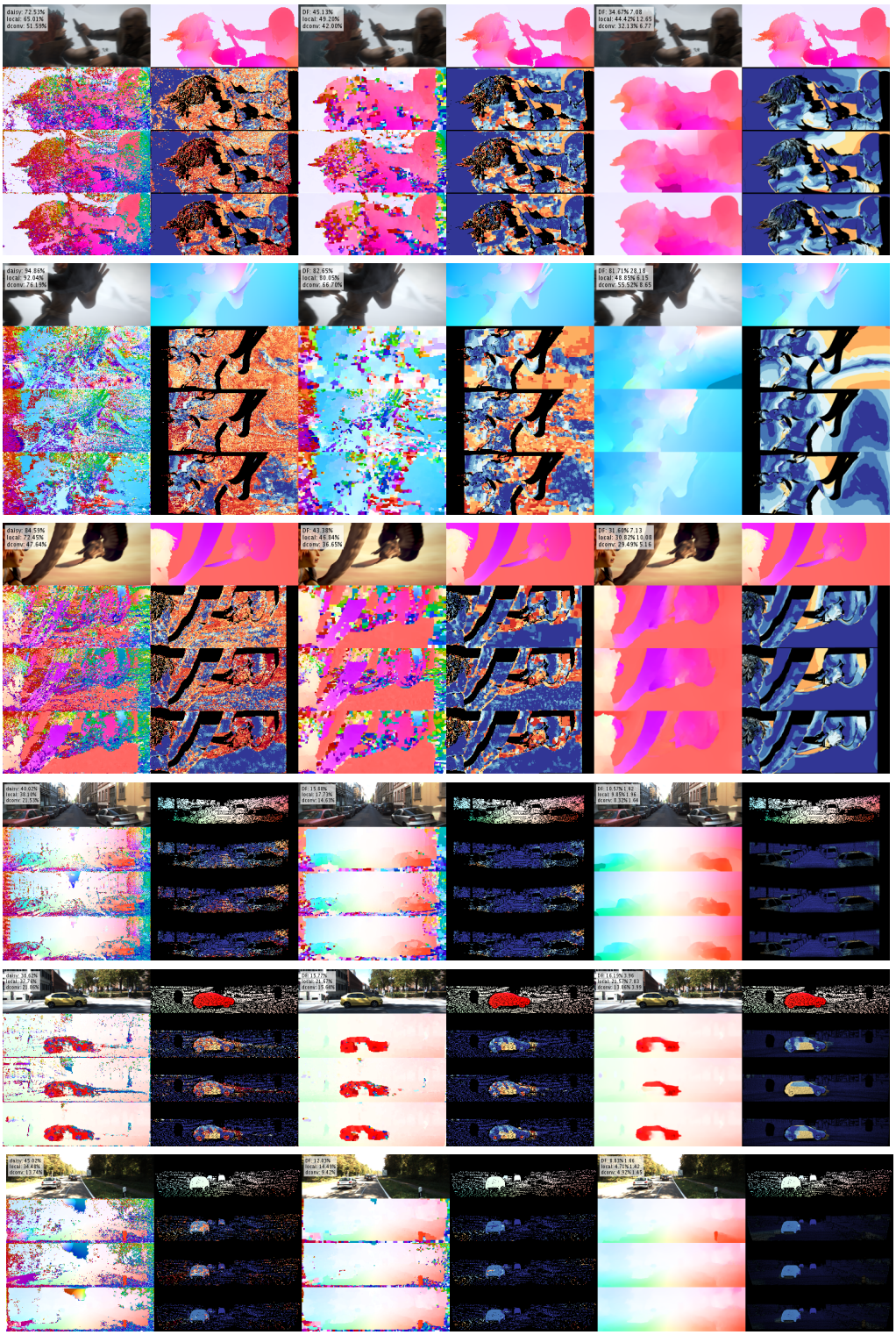


Figure 6.5: **Qualitative Results.** See Section 6.3.8 for details.

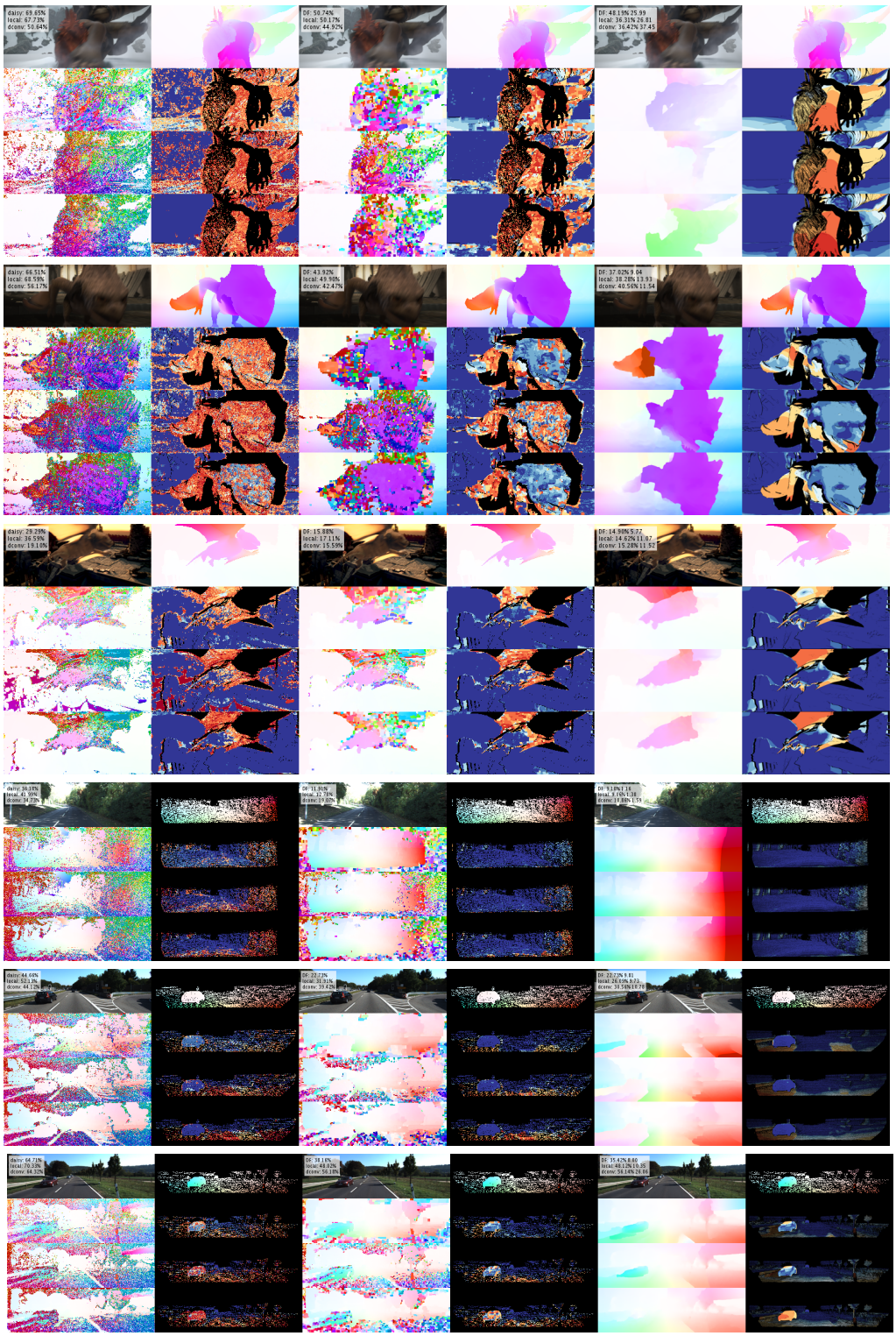


Figure 6.6: **Qualitiative Results.** See Section 6.3.8 for details.

Local Arch.	Run-time
1	1.27
2	1.89
3	2.08
4	2.52
5	2.62

Context Arch.	Run-time	Context Arch.	Run-time
1	1.06	11	1.63
2	0.76	12	0.87
3	0.45	13	0.46
4	0.76	14	0.88
5	0.46	15	0.46
6	0.45	16	0.51
7	1.06	17	1.13
8	1.67	18	2.10
9	1.06	19	1.46

Table 6.9: **Runtime (in sec) of Different Local and Context Architectures** This table compares the runtime of forward propagating two full-resolution Sintel images using different architectures.

by the proposed strided convolution implementation. We measure the average runtime of forward and backward propagation of a batch of 128 patch pairs followed by normalization and similarity score computation. As evidenced by our experiments, the proposed implementation leads to runtime gains up to two orders of magnitude compared to the naïve implementation. In other words, using the naïve implementation would require 200 days for training a model which we train in 2 days using the proposed implementation for some architectures.

Runtime Comparison of Different Architectures

In Table 6.9, we compare the runtime of different local and context architectures for one forward pass of two full-resolution images. For context, we only report the additional time after the local architecture 1. Note that this number would be different depending on the number of feature maps in the last layer of the local architecture. Using the context architecture 12 on top of local architecture 1, runtime is approximately 2 seconds in total.

Runtime of Full Deep Discrete Flow Model

In Section 6.3.9, we compared the runtime of different architectures. In this section, we fix the local and context architectures to 1 and 12, respectively, and perform a runtime analysis of the following stages in our pipeline. In Table 6.10, we list the runtime for different parts of the discrete inference together with the post-processing and the interpolation. Since these measurements are image dependent, we take the average runtime per stage over a set of images in the validation set of each dataset.

We precompute the pairwise term using parallelism and hashing as in Menze *et al.* (2015), and use an efficient implementation of Belief Propagation (BP) based on parallel checker-board update scheme. The parallel BP has a comparable runtime to the Block

	Refining Data Term	Pre-computing Pairwise	Belief Propagation	Post-processing	Interpolation
MPI Sintel	49.97	10.13	5.24	0.002	2.20
KITTI	35.42	14.17	10.54		1.66

Table 6.10: **Runtime (in sec) Analysis of Individual Stages** This table lists the runtime for different parts of the discrete inference as average over a set of validation images.

Parameter	DF Menze *et al.* (2015)	Optimized
Radius	5	10
Radius Quantization No.	4	4
Angular Quantization No.	4	8
Histogram Quantization No.	4	8
Normalization Type	2	2

Table 6.11: **Feature Matching Baseline Parameters.** Here, we show the values of the parameters from Discrete Flow (Menze *et al.*, 2015) and optimized for the feature matching baseline.

Coordinate Descent used in Menze *et al.* (2015) which takes 20 to 25 seconds in total for inference including the pairwise computation. As shown in Table 6.10, the slowest part is refining the data term using non-maxima suppression and adding proposals from the neighboring pixels. The post-processing and interpolation runtimes are negligible compared to the runtime of the discrete inference algorithm.

6.3.10 Parameter Settings

In this section, we provide the details of the parameter settings used to produce the reported results. Specifically, we list the parameters used in the feature matching baseline, discrete optimization, and the post-processing.

Table 6.11 lists the parameters of the Daisy feature descriptor used as baseline in Section 6.3.1. The first column corresponds to the parameters used in Discrete Flow (Menze *et al.*, 2015) and the second column to the optimized parameters on the MPI Sintel.

Table 6.12 shows the parameters of the discrete optimization and the post-processing. We optimized over these parameters using BCD on the validation sets, to minimize EPE in case of MPI Sintel and the average outlier ratio in case of KITTI as evaluated by the benchmarks. We empirically also tested larger values for the size of the label set by keeping the ratio of number of proposals from neighbors fixed, but did not observe a significant change in the performance, therefore we fixed the size of the label set to 300 during the optimization. In addition, we kept the NMS radius and the stride fixed. We observed that more iterations or a larger truncation threshold of the pairwise term give

Parameter	Value
Number of Initial Matches	1024
Size of the Label Set	300
Proposals from Neighbours	210
NMS Radius in Pixels	2
Stride for Discrete CRF in Pixels	4
Truncation Threshold of the Data Term	-0.25
Truncation Threshold of the Pairwise Term	15
Relative Weight of Pairwise Term	0.009
Number of Iterations	10

(a)

Discrete Optimization Parameters

Parameter	MPI Sintel	KITTI
Similarity Threshold (fw-/bw-check)	7.63	2.83
Minimum Segment Size (outlier removal)	100	277.45
Consistency Threshold (outlier removal)	5	13.01
Epic Flow	dataset	

(b)

Post-processing Parameters

Table 6.12: **Model Parameters.** Here, we list the parameter settings used for discrete optimization in (a), and for post-processing and interpolation in (b).

slightly better results, but at the cost of higher runtime. For interpolation, we used the default values as defined in Epic Flow for each dataset.

Chapter 7

Conclusions and Outlook

In this thesis, we investigated the common sources of error for matching-based problems. We proposed non-local solutions which leverage semantics and contextual information. For binocular stereo estimation, we follow the planarity-based approaches using superpixels, but go beyond local interactions by relating superpixels over large distances. A strong semantic prior is formed by locating objects and reasoning about their shape. For multi-view reconstruction, we use a volumetric depth-fusion approach based on the TSDF representation, but jointly model structural dependencies between different instances of objects. Semantics in 3D is exploited explicitly to detect instances and implicitly to cluster and regularize over objects of similar shape. For optical flow estimation, we follow the discrete flow framework by replacing the hand-crafted features with a learning-based approach to create proposals. Contextual information is incorporated using dilated convolutions to increase the receptive field size efficiently. In each case, our experiments demonstrate the importance of extending traditional approaches to model non-local interactions by leveraging semantics and context.

In this section, we summarize our approach and our findings for each proposed solution in the thesis. We emphasize the advantages of using high-level cues for matching-based problems and point out remaining challenges as well as possible future directions.

We proposed displets as an expressive non-local prior for resolving ambiguities in stereo matching. By conditioning on the semantic class of an object and an initial depth estimation, we sample likely configurations of 3D CAD models in the scene. Displets provide valuable category specific shape information which allows for regularizing the solution over very large distances in the image. Superpixels inside an object mask are encouraged to take orientations suggested by the sampled displets for that region. The experiments show significant decreases in error in reflective and textureless regions which is also reflected in the overall results. The performance still improves even when using a smaller number of different CAD models. This finding supports our initial motivation for modeling the shape of cars which are geometrically well-defined objects. Similarly, limiting the number of displet proposals affects the performance but still produces reasonable results. In general, the performance is not very sensitive with respect to small parameter changes.

We presented a novel method for jointly reconstructing objects with similar shapes

in 3D by optimizing their pose and shape parameters using a volumetric representation. A challenging dataset is recorded from moving omnidirectional cameras under regular daylight conditions. After localizing and calibrating cameras, an initial volumetric representation is obtained by fusing depth maps estimated from spherically rectified stereo pairs. The challenges of the dataset are evidently pronounced in the initial reconstruction such as the missing walls of buildings and the erroneous reconstruction of cars. We overcome these challenges by first detecting object instances and then learning shape models of similar objects. This is achieved by formulating an objective function enforcing agreement in shape as well as scale over different observations associated to a model. Our joint formulation results in increased performance compared to baselines including our initial reconstruction. Our method improves in particular with respect to completeness as it transfers surface knowledge between objects of similar shape. Qualitative results including visualizations of the accuracy and the completeness show that completed surfaces do not compromise accuracy.

We presented an efficient way of learning context-aware features for optical flow in a discrete framework by showing that dilated convolutions can be implemented efficiently for patch-based training. A local network with regular convolutions followed by a context network with dilated convolutions are trained for feature matching. The learned representations are used to create the proposal set for the discrete optimization framework. Exact matching can be performed efficiently on the GPU, resulting in significantly lower outlier ratios compared to an approximate matching method. We performed extensive evaluations on different local and context architectures using Daisy features and previously proposed architectures as baselines. Learning features with context networks improves feature matching performance with respect to local architectures and Daisy features on both the MPI Sintel and the KITTI datasets. Although the improvements are less pronounced in later stages, the learned feature representations with context can better handle challenging situations such as textureless regions and repetitive structures.

High runtime is a common weakness of our approaches. The runtime in the order of minutes is not suitable for applications such as autonomous driving. End-to-end deep learning methods provide a promising direction towards reducing runtime. In addition, our 3D reconstruction model has limited robustness in detecting outliers. In the future, we thus plan to incorporate outlier handling by optimizing a robust function. A related problem in shape modeling is that the simple mean shape produces better results than using an additional principal component due to the stronger regularization required. Improving the noise handling in the initial reconstruction or a more robust shape model could potentially alleviate this problem. Although our experiments demonstrated that learning features with context is crucial for reducing outliers in the WTA solution of the network, large gains mostly diminish in the later stages of the pipeline. We found that large changes in scale pose problems to current feature learning approaches, prompting for the development of inherently scale invariant deep features. We also remark that the performance of deep discrete flow is hampered by the piece-wise training. We therefore plan to investigate end-to-end training by back-propagating errors through all stages of

our pipeline.

In binocular stereo estimation, we deliberately focused on the most problematic object category, but displets can be applied to other geometrically well constrained object classes as well. Buildings, for instance, often lack texture but their shape can be well described by a set of planes or 3D box primitives. We included buildings in the multi-view case and we note that the joint reconstruction can be extended to even more object categories. Furthermore, it can be combined with external 3D shape knowledge similar to displets. A joint model can be learned from 3D CAD models as well as a collection of automatically retrieved objects in large scenes. We didn't explicitly model semantics for optical flow, but enabled the network to model nonlocal interactions from a large context which contain useful information for matching. An interesting future direction is the extension of displets to flowlets for serving as nonlocal category specific prior in optical flow and scene flow. In addition to flowlets, background motion modeling can be integrated into discrete flow framework as a strong prior on the optical flow due to the camera motion. This could potentially improve problems in very large regions on the image such as the challenging snow scenes on the MPI Sintel with very little texture or driving sequences on the KITTI with large scale changes due to the ego-motion of the vehicle.

We considered visual perception from the perspective of low-level methods by incorporating our prior knowledge about the semantics into the process. We identified frequently observed objects in street scenes that are challenging to reconstruct for current methods. We located them on the image or the reconstruction and jointly modeled their shape and the 3D configuration in our formulations. Finally, we proposed an efficient feature learning method which enables non-local representations for optical flow and possibly other feature matching tasks. Our joint formulations show significant improvements over the baselines, especially in problematic regions. Examples include the reflective surfaces of cars, textureless walls of buildings or snow scenes which are very challenging for the previous methods. We think that using semantics and a larger context is a very natural way of handling these commonly observed ambiguities towards achieving the end-goal of brain performance. Despite the high runtime and the other problems, this thesis takes an important step in bridging the gap between different levels of reasoning in computer vision with many possible extensions and improvements.

Contributions

The work presented in this thesis mainly comprises the following three publications:

- Displets: Resolving Stereo Ambiguities using Object Knowledge. Güney, F., Geiger, A. In *CVPR* 2015.
- Exploiting Object Similarity in 3D Reconstruction. Zhou, C., Güney, F., Wang, Y., Geiger, A. In *ICCV* 2015.
- Deep Discrete Flow. Güney, F., Geiger, A. Geiger, A. In *ACCV* 2016.

During my PhD, I also worked on the following publications which are not the main focus of this thesis:

- Slow Flow: Exploiting High-Speed Cameras for Accurate and Diverse Optical Flow Reference Data. Janai, J., Güney, F., Wulff, J., Black, M. J., Geiger, A. In *CVPR* 2017.
- Computer Vision for Autonomous Vehicles: Problems, Datasets and State-of-the-Art. Janai, J., Güney, F., Behl, A., Geiger, A. In *Arxiv*, 2017.

I declare that this thesis has been composed by me based on these publications. My advisor, Andreas Geiger, has been involved in all the projects and contributed ideas, text, code and illustrative figures. The foundations chapter (Section 2) includes an adaptation of some sections from the survey preprint. The 3D Reconstruction chapter (Section 5) contains work and results based on the code and the figures provided by my coauthor Chen Zhou. The other sources used including the data, code and the figures adapted are referenced in the text. All the experiments in this work are the result of my own work unless otherwise stated.

Appendix A

3D Reconstruction

A.1 BCD Inference

This section provides details on optimizing the individual blocks in our objective function Eq. 5.21:

$$\underset{\pi,\phi,\mathbf{X},\mathbf{V},\mathbf{k}}{\operatorname{argmin}} \sum_{i=1}^{N} \Psi_i(\pi,\phi,\mathbf{X},\mathbf{V},\mathbf{k}) \tag{A.1}$$

with the corresponding energy

$$\Psi_i(\cdot) = \psi_{shp}(\pi_i,\phi_{k_i},\mathbf{x}_i) + \lambda_{scale}\,\psi_{scale}(\mathbf{s}_i,\mathbf{v}_{k_i}) + \lambda_{reg}\,\psi_{reg}^{(i)}(\pi_i) \tag{A.2}$$

A.1.1 Object Poses and Model Scales

The first block optimizes the object poses π jointly with the model scales $\mathbf{V}$ while keeping the other variables fixed. Due to the small number of parameters involved, we leverage gradient descent in order to find a local minimum. We first differentiate the objective function in Eq. A.1 with respect to $\{\mathbf{t}_i, r_i, \mathbf{s}_i\}$ and $\mathbf{V}$ as shown below, and then solve the non-linear least squares problem using the Ceres solver Agarwal *et al.* (2010).

For clarity, we consider a single term Ψ_i by dropping the observation index i in the following. First, we note that the derivatives of $\psi_{scale}(\mathbf{s},\mathbf{v}_k)$ and $\psi_{reg}(\pi)$ with respect to $\{\pi,\mathbf{V}\}$ are readily given due to their simple quadratic form. The derivative of $\psi_{shp}(\pi,\phi,\mathbf{x})$ with respect to π:

$$\begin{aligned}\frac{\partial \psi_{shp}(\pi,\phi,\mathbf{x})}{\partial \pi} \;=\; & \sum_{\mathbf{p}\in\Omega} \left(\phi(\mathbf{p},\mathbf{x}) - d(\pi(\mathbf{p}))\right)^2 \frac{\partial w}{\partial \pi} - \\ & 2w(\pi(\mathbf{p}))\,\left(\phi(\mathbf{p},\mathbf{x}) - d(\pi(\mathbf{p}))\right) \frac{\partial d}{\partial \pi}\end{aligned} \tag{A.3}$$

Let $\mathbf{p}_t$ denote the the transformed point $\pi(\mathbf{p}) \in \mathbb{R}^3$. By using the chain rule, we obtain

the following:

$$\frac{\partial w}{\partial \pi} = \frac{\partial w}{\partial \mathbf{p}_t}\,\frac{\partial \mathbf{p}_t}{\partial \pi} \qquad\qquad \frac{\partial d}{\partial \pi} = \frac{\partial d}{\partial \mathbf{p}_t}\,\frac{\partial \mathbf{p}_t}{\partial \pi} \tag{A.4}$$

Gradients of the weight w and the signed distance function d with respect to $\mathbf{p}_t$ can be computed using bilinear interpolation. According to our parametrization, the transformed point is given by

$$\mathbf{p}_t = \mathbf{RSp} + [\mathbf{t}\ 0]^T \qquad\qquad \mathbf{R} = \begin{bmatrix} \cos r & -\sin r & 0 \\ \sin r & \cos r & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.5}$$

where $\mathbf{R}$ is the rotation matrix formed by 2D rotation angle r, $\mathbf{S}$ is the diagonal scaling matrix, represented by the vector $\mathbf{s}$, and $\mathbf{t}$ is the 2D translation vector in the x-y plane, i.e. the ground plane. Thus, we compute the gradient of $\mathbf{p}_t$ with respect to π in terms of r, $\mathbf{s}$, and $\mathbf{t}$ as follows:

$$\begin{aligned} \frac{\partial \mathbf{p}_t}{\partial r} &= \frac{\partial(\mathbf{RSp}+\mathbf{t})}{\partial r} \\ &= \begin{bmatrix} -\sin(r)s_x p_x - \cos(r)s_y p_y \\ \cos(r)s_x p_x - \sin(r)s_y p_y \\ 0 \end{bmatrix} \end{aligned} \tag{A.6}$$

$$\begin{aligned} \frac{\partial \mathbf{p}_t}{\partial \mathbf{s}} &= \frac{\partial(\mathbf{RSp}+\mathbf{t})}{\partial \mathbf{s}} \\ &= \begin{bmatrix} \cos(r)p_x & -\sin(r)p_y & 0 \\ \sin(r)p_x & \cos(r)p_y & 0 \\ 0 & 0 & p_z \end{bmatrix} \end{aligned} \tag{A.7}$$

$$\frac{\partial \mathbf{p}_t}{\partial \mathbf{t}} = \begin{bmatrix} \mathbf{I}_{2\times 2} \\ \mathbf{0}_{1\times 2} \end{bmatrix} \tag{A.8}$$

With other variables fixed, the terms including $\mathbf{V}$ are also simple quadratic terms with trivial derivatives.

A.1.2 Coefficients and Model Associations

The final block optimizes the coefficients $\mathbf{X}$ jointly with the model associations $\mathbf{k}$ while keeping the other variables fixed. We first note that this optimization can be performed independently for each observation $i \in \{1,\ldots,N\}$. We thus obtain $\mathbf{x}_i$ and k_i for observa-

tion i as

$$\underset{\mathbf{x}_i, k_i}{\operatorname{argmin}} \; \psi_{shp}(\pi_i, \phi_{k_i}, \mathbf{x}_i) + \lambda_{scale}\, \psi_{scale}(\mathbf{s}_i, \mathbf{v}_{k_i}) \tag{A.9}$$

which can be found by minimization with respect to $\mathbf{x}_i$ for each $k_i \in \{1, \ldots, M\}$.

With slight change in notation, let the linear model associated with observation i be described by the mean $\mu_{k_i} \in \mathbb{R}^{|\Omega|}$ and the D-dimensional orthonomal basis $\xi_{k_i}^d \in \mathbb{R}^{|\Omega|}$ for $d \in \{1, \ldots, D\}$. The coefficient vector $\mathbf{x}_i$ for observation i is obtained by solving the following weighted least squares estimation problem in closed form:

$$\mathbf{B}_{k_i}^T \mathbf{W}_i \mathbf{B}_{k_i} \mathbf{x}_i = \mathbf{B}_{k_i} \mathbf{W}_i (\mathbf{d}_i - \mu_{k_i}) \tag{A.10}$$

Here, $\mathbf{B} = \{\xi_{k_i}^1, \ldots, \xi_{k_i}^D\}$ denotes the $|\Omega| \times D$ basis matrix, $\mathbf{W}_i = \operatorname{diag}(w(\pi_i(\mathbf{p}))) \in \mathbb{R}^{|\Omega| \times |\Omega|}$, and $\mathbf{d}_i = d(\pi_i(\mathbf{p})) \in \mathbb{R}^{|\Omega|}$ is the vector of signed distance function values for observation i. We estimate the solution to Eq. A.10 for each $k_i \in \{1, \ldots, M\}$ in order to find the minimizer of Eq. A.9.

Symbols

We list the notation and symbols which are common across multiple chapters in the thesis. The specific notations are introduced in each chapter.

General Notation

Scalars	Regular lower case	f, z, σ
Vectors	Bold lower case	$\mathbf{f}$, $\mathbf{x}$, $\mathbf{d}$
Matrices	Bold upper case	$\mathbf{P}$, $\mathbf{F}$, $\mathbf{E}$
Sets	Calligraphic upper case	$\mathcal{D}$, $\mathcal{L}$, $\mathcal{S}$
Distributions	Calligraphic upper case	$\mathcal{U}(\cdot)$, $\mathcal{N}(\cdot)$

$\mathbb{R}$	Real numbers
x_i	i'th element of vector $\mathbf{x}$
$x^{(i)}$	i'th sample in time
$A_{i,j}$	(i,j)'th element of matrix $\mathbf{A}$
$p(\cdot)$	Probability
$\langle \cdot, \cdot \rangle$	Inner product
$[\cdot]$	Iverson bracket (1 if true, 0 otherwise)
$\delta(\cdot, \cdot)$	Kronecker delta (1 if arguments are equal, 0 otherwise)

Abbreviations

AAE	Average Angular Error
ANN	Approximate Nearest Neighbor
BCD	Block Coordinate Descent
BP	Belief Propagation
CAD	Computer-Aided Design
CCD	Charge-Coupled Device
CMOS	Complementary Metal-Oxide Semiconductor sensors
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CRF	Conditional Random Field
DF	Discrete Flow
DSI	Disparity Space Image
EPE	End-Point Error
FLANN	Fast Library for Approximate Nearest Neighbors
GPS	Global Positioning System
GPU	Graphical Processing Unit
GP-LVM	Gaussian Process Latent Variable Model
HMI	Hierarchical Mutual Information
IMU	Inertial Measurement Unit
KDE	Kernel Density Estimation
KL	Kullback-Leibler
LIDAR	Light Detection and Ranging
LoG	Laplacian of Gaussian
MAP	Maximum A Posteriori
MCMC	Markov Chain Monte Carlo
MDL	Minimum Description Length
MP-PBP	Max Product Particle Belief Propagation
MRF	Markov Random Field
MVS	Multi-View Stereo
MWG	Metropolis-Within-Gibbs
NCC	Normalized Cross Correlation
NMS	Non-Maximum Suppression
NP	Non-Deterministic Polynomial
PBP	Particle Belief Propagation

PC	Principle Component
PCA	Principle Component Analysis
PMVS	Patch-based Multi-View Stereo
PWP	Pixel-wise Posterior
P3P	Perspective-Three-Point
RANSAC	Random Sample Consensus
ReLU	Rectified Linear Unit
RFS	Receptive Field Size
RGB	Red, Green and Blue
RGB-D	Red, Green, Blue - Depth
RMS	Root Mean Squared
SfM	Structure from Motion
SGM	Semi-Global Matching
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SOR	Successive Over-Relaxation
SSD	Sum of Squared Differences
SVM	Support Vector Machine
TRW	Tree-Reweighted
TRW-S	Tree-Reweighted Sequential
TSDF	Truncated Signed Distance Value
TV	Total Variation
TGV	Total Generalized Variation
UV	Ultra Violet
VO	Visual Odometry
V1	Visual area 1 in the visual cortex of the brain
V2	Visual area 2 in the visual cortex of the brain
WebGL	Web Graphics Library
WTA	Winner Takes All
nD	n Dimensional

Bibliography

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **34**(11), 2274–2282.

Agarwal, S., Mierle, K., and Others (2010). Ceres solver. `http://ceres-solver.org`.

Alvarez, L., Weickert, J., and Sánchez, J. (2000). Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision (IJCV)*, **39**(1), 41–56.

Anandan, P. (1989). A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision (IJCV)*, **2**(3), 283–310.

Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, **50**(1-2), 5–43.

Bai, M. and Urtasun, R. (2016). Deep watershed transform for instance segmentation. *arXiv.org*, **1611.08303**.

Bai, M., Luo, W., Kundu, K., and Urtasun, R. (2016). Exploiting semantic information and deep matching for optical flow. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Bailer, C., Taetz, B., and Stricker, D. (2015). Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Bailer, C., Varanasi, K., and Stricker, D. (2017). Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Baker, S. and Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision (IJCV)*, **56**(3), 221–255.

Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M., and Szeliski, R. (2011). A database and evaluation methodology for optical flow. *International Journal of Computer Vision (IJCV)*, **92**, 1–31.

Bao, L., Yang, Q., and Jin, H. (2014). Fast edge-preserving PatchMatch for large displacement optical flow. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Bao, S., Chandraker, M., Lin, Y., and Savarese, S. (2013). Dense object reconstruction with semantic priors. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.

Barnes, C., Shechtman, E., Finkelstein, A., and Goldman, D. B. (2009). Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. on Graphics (SIGGRAPH)*, **28**(3), 24–1.

Besse, F., Rother, C., Fitzgibbon, A., and Kautz, J. (2014). PMBP: PatchMatch Belief Propagation for correspondence field estimation. *International Journal of Computer Vision (IJCV)*, **110**(1), 2–13.

Bhotika, R., Fleet, D. J., and Kutulakos, K. N. (2002). A probabilistic theory of occupancy and emptiness. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Bibby, C. and Reid, I. (2008). Robust real-time visual tracking using pixel-wise posteriors. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Biederman, I. (1987). Recognition-by-components: a theory of human image understanding. *Psychological Review*, **94**(2), 115.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, 1st ed. 2006 edition.

Black, M. J. and Anandan, P. (1993). A framework for the robust estimation of optical flow. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Black, M. J. and Anandan, P. (1996). The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding (CVIU)*, **63**(1), 75–104.

Black, M. J. and Jepson, A. D. (1996). Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **18**(10), 972–986.

Black, M. J. and Yacoob, Y. (1997). Recognizing facial expressions in image sequences using local parameterized models of image motion. *International Journal of Computer Vision (IJCV)*, **25**(1), 23–48.

Blaha, M., Vogel, C., Richard, A., Wegner, J. D., Pock, T., and Schindler, K. (2016). Large-scale semantic 3d reconstruction: An adaptive multi-resolution model for multi-class volumetric labeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Bleyer, M., Rother, C., and Kohli, P. (2010). Surface stereo with soft segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Bleyer, M., Rother, C., Kohli, P., Scharstein, D., and Sinha, S. (2011). Object stereo - joint stereo matching and object segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Bleyer, M., Rhemann, C., and Rother, C. (2012). Extracting 3D scene-consistent object proposals and depth from stereo images. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Bredies, K., Kunisch, K., and Pock, T. (2010). Total generalized variation. *Journal of Imaging Sciences (SIAM)*, **3**(3), 492–526.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a siamese time delay neural network. In *Advances in Neural Information Processing Systems (NIPS)*.

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov Chain Monte Carlo*. CRC press.

Brox, T. and Malik, J. (2011). Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **33**, 500–513.

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). A naturalistic open source movie for optical flow evaluation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Calakli, F. and Taubin, G. (2011). SSD: smooth signed distance surface reconstruction. *Computer Graphics Forum*, **30**(7), 1993–2002.

Cao, Y., Wu, Z., and Shen, C. (2016). Estimating depth from monocular images as classification using deep fully convolutional residual networks. *CoRR*, **abs/1605.02305**.

Carreira, J. and Sminchisescu, C. (2012). CPMC: automatic object segmentation using constrained parametric min-cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **34**(7), 1312–1328.

Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2015a). Semantic image segmentation with deep convolutional nets and fully connected crfs. In *Proc. of the International Conf. on Learning Representations (ICLR)*.

Chen, Q. and Koltun, V. (2016). Full flow: Optical flow estimation by global optimization over regular grids. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Chen, Z., Sun, X., Wang, L., Yu, Y., and Huang, C. (2015b). A deep visual correspondence embedding model for stereo matching costs. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Cherabier, I., Häne, C., Oswald, M. R., and Pollefeys, M. (2016). Multi-label semantic 3d reconstruction using voxel blocks. In *Proc. of the International Conf. on 3D Vision (3DV)*.

Collins, R. T. (1996). A space-sweep approach to true multi-image matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 358–363.

Cornelis, N., Leibe, B., Cornelis, K., and Van Gool, L. J. (2008). 3D urban scene modeling integrating recognition and reconstruction. *International Journal of Computer Vision (IJCV)*, **78**(2-3), 121–141.

Cremers, D. and Soatto, S. (2005). Motion competition: A variational approach to piecewise parametric motion segmentation. *International Journal of Computer Vision (IJCV)*, **62**(3), 249–265.

Cremers, D., Rousson, M., and Deriche, R. (2007). A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *International Journal of Computer Vision (IJCV)*, **72**, 215.

Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. In *ACM Trans. on Graphics (SIGGRAPH)*.

Dame, A., Prisacariu, V., Ren, C., and Reid, I. (2013). Dense reconstruction using 3D object shape priors. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Davis, J., Marschner, S. R., Garr, M., and Levoy, M. (2002). Filling holes in complex surfaces using volumetric diffusion. In *Proc. of the International Conf. on 3D Digital Imaging, Modeling, Data Processing, Visualization and Transmission (THREEDIM-PVT)*.

Deng, W., Hu, J., Lu, J., and Guo, J. (2014). Transform-invariant PCA: A unified approach to fully automatic facealignment, representation, and recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **36**(6), 1275–1284.

Dosovitskiy, A., Fischer, P., Ilg, E., Haeusser, P., Hazirbas, C., Golkov, V., v.d. Smagt, P., Cremers, D., and Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Drory, A., Haubold, C., Avidan, S., and Hamprecht, F. A. (2014). Semi-global matching: A principled derivation in terms of message passing. In *Proc. of the German Conference on Pattern Recognition (GCPR)*.

Eigen, D. and Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*.

Faugeras, O. D. and Keriven, R. (1998). Variational principles, surface evolution, pdes, level set methods, and the stereo problem. *IEEE Trans. on Image Processing (TIP)*, **7**(3), 336–344.

Fei-Fei, L., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **28**(4), 594–611.

Felzenszwalb, P. and Huttenlocher, D. (2006). Efficient belief propagation for early vision. *International Journal of Computer Vision (IJCV)*, **70**(1), 41–54.

Fleet, D. J., Black, M. J., Yacoob, Y., and Jepson, A. D. (2000). Design and use of linear models for image motion analysis. *International Journal of Computer Vision (IJCV)*, **36**(3), 171–193.

Fouhey, D. F., Gupta, A., and Hebert, M. (2013). Data-driven 3D primitives for single image understanding. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Frahm, J.-M., Fite-Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.-H., Dunn, E., Clipp, B., Lazebnik, S., and Pollefeys, M. (2010). Building rome on a cloudless day. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Fuhrmann, S. and Goesele, M. (2011). Fusion of depth maps with multiple scales. In *ACM Trans. on Graphics*, volume 30, page 148. ACM.

Fuhrmann, S., Ackermann, J., Kalbe, T., and Goesele, M. (2010). Direct resampling for isotropic surface remeshing. In *Vision, Modeling and Visualization (VMV)*.

Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **32**(8), 1362–1376.

Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R. (2009). Manhattan-world stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Gadot, D. and Wolf, L. (2016). Patchbatch: a batch augmented loss for optical flow. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Gallup, D., Frahm, J.-M., and Pollefeys, M. (2010a). A heightmap model for efficient 3d reconstruction from street-level video. In *Proc. of the DAGM Symposium on Pattern Recognition (DAGM)*, volume 6.

Gallup, D., Frahm, J.-M., and Pollefeys, M. (2010b). Piecewise planar and non-planar stereo for urban scene reconstruction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Gálvez-López, D. and Tardós, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Trans. on Robotics*, **28**(5), 1188–1197.

Garland, M. and Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In *ACM Trans. on Graphics (SIGGRAPH)*, pages 209–216.

Gehrig, S. K., Eberli, F., and Meyer, T. (2009). A real-time low-power stereo vision engine using semi-global matching. In *Proc. of the International Conf. on Computer Vision Systems (ICVS)*.

Geiger, A., Roser, M., and Urtasun, R. (2010). Efficient large-scale stereo matching. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, **32**(11), 1231–1237.

Geyer, C. and Daniilidis, K. (2000). A unifying theory for central panoramic systems and practical implications. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Goesele, M., Snavely, N., Curless, B., Hoppe, H., and Seitz, S. M. (2007). Multi-view stereo for community photo collections. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Güney, F. and Geiger, A. (2015). Displets: Resolving stereo ambiguities using object knowledge. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Güney, F. and Geiger, A. (2016). Deep discrete flow. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*.

Han, X., Leung, T., Jia, Y., Sukthankar, R., and Berg, A. C. (2015). Matchnet: Unifying feature and metric learning for patch-based matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Häne, C., Zach, C., Zeisl, B., and Pollefeys, M. (2012). A patch prior for dense 3d reconstruction in man-made environments. In *Proc. of the International Conf. on 3D Digital Imaging, Modeling, Data Processing, Visualization and Transmission (THREEDIMPVT)*.

Häne, C., Zach, C., Cohen, A., Angst, R., and Pollefeys, M. (2013). Joint 3D scene reconstruction and class segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Häne, C., Savinov, N., and Pollefeys, M. (2014a). Class specific 3d object shape priors using surface normals. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Häne, C., Heng, L., Lee, G. H., Sizov, A., and Pollefeys, M. (2014b). Real-time direct dense matching on fisheye images using plane-sweeping stereo. In *Proc. of the International Conf. on 3D Vision (3DV)*.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

Heng, L., Li, B., and Pollefeys, M. (2013). Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*.

Heng, L., Bürki, M., Lee, G. H., Furgale, P., Siegwart, R., and Pollefeys, M. (2014). Infrastructure-based calibration of a multi-camera rig. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Heng, L., Furgale, P. T., and Pollefeys, M. (2015). Leveraging image-based localization for infrastructure-based calibration of a multi-camera rig. *Journal of Field Robotics (JFR)*, **32**(5), 775–802.

Hirschmüller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(2), 328–341.

Hirschmüller, H. and Scharstein, D. (2007). Evaluation of cost functions for stereo matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Hoiem, D., Efros, A. A., and Hebert, M. (2005). Geometric context from a single image. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Horn, B. K. P. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence (AI)*, **17**(1-3), 185–203.

Hornacek, M., Rhemann, C., Gelautz, M., and Rother, C. (2013). Depth super resolution by rigid body self-similarity in 3d. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Hornacek, M., Besse, F., Kautz, J., Fitzgibbon, A. W., and Rother, C. (2014). Highly overparameterized optical flow using PatchMatch Belief Propagation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Hosni, A., Bleyer, M., Gelautz, M., and Rhemann, C. (2009). Local stereo matching using geodesic support weights. In *Proc. IEEE International Conf. on Image Processing (ICIP)*.

Howard, I. and Rogers, D. (2008). *Seeing in Depth: Volume 1: Basic Mechanics/ Volume 2: Depth Perception 2-Volume Set*. Oxford University Press, USA.

Ihler, A. and McAllester, D. (2009). Particle belief propagation. In *Conference on Artificial Intelligence and Statistics (AISTATS)*.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2016). Flownet 2.0: Evolution of optical flow estimation with deep networks. *arXiv.org*, **1612.01925**.

Jacquet, B., Hane, C., Koser, K., and Pollefeys, M. (2013). Real-world normal map capture for nearly flat reflective surfaces. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Ju, S. X., Black, M. J., and Jepson, A. D. (1996). Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Julesz, B. (1971). *Foundations of Cyclopean Perception*. University of Chicago Press.

Karpathy, A., Miller, S., and Fei-Fei, L. (2013). Object discovery in 3d scenes via shape analysis. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Karsch, K., Liu, C., and Kang, S. B. (2014). Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **36**(11), 2144–2158.

Kazhdan, M. M. and Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Trans. on Graphics (SIGGRAPH)*, **32**(3), 29.

Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., and Bry, A. (2017). End-to-end learning of geometry and context for deep stereo regression. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Klaus, A., Sormann, M., and Karner, K. (2006). Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Proc. of the International Conf. on Pattern Recognition (ICPR)*.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *Proc. of the International Symposium on Mixed and Augmented Reality (ISMAR)*.

Klingner, B., Martin, D., and Roseborough, J. (2013). Street view motion-from-structure-from-motion. In *The IEEE International Conference on Computer Vision (ICCV)*.

Kneip, L., Scaramuzza, D., and Siegwart, R. (2011). A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Kolmogorov, V. (2006). Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **28**(10), 1568–1583.

Komodakis, N. and Paragios, N. (2009). Beyond pairwise energies: Efficient optimization for higher-order MRFs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in Neural Information Processing Systems (NIPS)*.

Krähenbühl, P. and Koltun, V. (2014). Geodesic object proposals. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Kundu, A., Li, Y., Dellaert, F., Li, F., and Rehg, J. (2014). Joint semantic segmentation and 3d reconstruction from monocular video. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Kuschk, G. and Cremers, D. (2013). Fast and accurate large-scale stereo reconstruction using variational methods. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*.

Kutulakos, K. N. and Seitz, S. M. (2000). A theory of shape by space carving. *International Journal of Computer Vision (IJCV)*, **38**(3), 199–218.

la Torre, F. D. and Black, M. J. (2001). Robust principal component analysis for computer vision. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, pages 362–369.

Labatut, P., Pons, J., and Keriven, R. (2007). Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, pages 1–8.

Labatut, P., Pons, J.-P., and Keriven, R. (2009). Hierarchical shape-based surface reconstruction for dense multi-view stereo. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*.

Ladicky, L., Sturgess, P., Russell, C., Sengupta, S., Bastanlar, Y., Clocksin, W., and Torr, P. (2010). Joint optimisation for object class segmentation and dense stereo reconstruction. In *Proc. of the British Machine Vision Conf. (BMVC)*.

Ladicky, L., Russell, C., Kohli, P., and Torr, P. H. S. (2014a). Associative hierarchical random fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **36**(6), 1056–1077.

Ladicky, L., Shi, J., and Pollefeys, M. (2014b). Pulling things out of perspective. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Lafarge, F. and Mallet, C. (2012). Creating large-scale city models from 3d-point clouds: A robust approach with hybrid representation. *International Journal of Computer Vision (IJCV)*, **99**(1), 69–85.

Lafarge, F., Descombes, X., Zerubia, J., and Deseilligny, M. P. (2010). Structural approach for building reconstruction from a single DSM. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **32**(1), 135–147.

Lafarge, F., Keriven, R., Bredif, M., and Vu, H.-H. (2013). A hybrid multiview stereo algorithm for modeling urban scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **35**(1), 5–17.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. In *Proc. of the International Conf. on 3D Vision (3DV)*, pages 239–248.

Lee, G. H., Fraundorfer, F., and Pollefeys, M. (2013). Structureless pose-graph loop-closure with a multi-camera system on a self-driving car. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*.

Li, Z. and Chen, J. (2015). Superpixel segmentation using linear spectral clustering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1356–1363.

Liu, C., Yuen, J., Torralba, A., Sivic, J., and Freeman, W. T. (2008). SIFT flow: Dense correspondence across different scenes. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Liu, F., Shen, C., Lin, G., and Reid, I. (2015). Learning depth from single monocular images using deep convolutional neural fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*.

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Trans. on Graphics (SIGGRAPH)*.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, **60**(2), 91–110.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proc. of the International Joint Conf. on Artificial Intelligence (IJCAI)*.

Luo, W., Schwing, A., and Urtasun, R. (2016a). Efficient deep learning for stereo matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Luo, W., Li, Y., Urtasun, R., and Zemel, R. S. (2016b). Understanding the effective receptive field in deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Ma, L. and Sibley, G. (2014). Unsupervised dense object discovery, detection, tracking and reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Malisiewicz, T., Gupta, A., and Efros, A. A. (2011). Ensemble of exemplar-svms for object detection and beyond. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Mansinghka, V., Kulkarni, T., Perov, Y., and Tenenbaum, J. (2013). Approximate bayesian image interpretation using generative probabilistic graphics programs. In *Advances in Neural Information Processing Systems (NIPS)*.

Marquez-Neila, P., Kohli, P., Rother, C., and Baumela, L. (2014). Non-parametric higher-order random fields for image segmentation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Marr, D. (1983). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Company.

Mayer, N., Ilg, E., Haeusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*.

Mei, C. and Rives, P. (2007). Single view point omnidirectional camera calibration from planar grids. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Menze, M. and Geiger, A. (2015). Object scene flow for autonomous vehicles. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Menze, M., Heipke, C., and Geiger, A. (2015). Discrete optimization for optical flow. In *Proc. of the German Conference on Pattern Recognition (GCPR)*.

Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.-M., Yang, R., Nistér, D., and Pollefeys, M. (2007). Real-time visibility-based fusion of depth maps. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Michael Bleyer, C. R. and Rother, C. (2011). Patchmatch stereo - stereo matching with slanted support windows. In *Proc. of the British Machine Vision Conf. (BMVC)*.

Micusik, B. and Kosecka, J. (2009). Piecewise planar city 3d modeling from street view panoramic sequences. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Muja, M. and Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **36**(11), 2227–2240.

Musialski, P., Wonka, P., Aliaga, D. G., Wimmer, M., Gool, L. J. V., and Purgathofer, W. (2013). A survey of urban reconstruction. *Computer Graphics Forum*, **32**(6), 146–177.

Nagel, H. (1987). On the estimation of optical flow: Relations between different approaches and some new results. *Artificial Intelligence (AI)*, **33**(3), 299–324.

Nakayama, K. and Shimojo, S. (1990). Da vinci stereopsis: Depth and subjective occluding contours from unpaired image points. *Vision Research*, **30**(11), 1811 – 1825. Optics Physiology and Vision.

Newcombe, R. A. and Andrew, J. D. (2010). Live dense reconstruction with a single moving camera davison. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Newcombe, R. A., Lovegrove, S., and Davison, A. J. (2011a). DTAM: dense tracking and mapping in real-time. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011b). Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. of the International Symposium on Mixed and Augmented Reality (ISMAR)*.

Nießner, M., Zollhöfer, M., Izadi, S., and Stamminger, M. (2013). Real-time 3d reconstruction at scale using voxel hashing. In *ACM Trans. on Graphics (SIGGRAPH)*.

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **26**(6), 756–777.

Noorshams, N. and Wainwright, M. J. (2013). Belief propagation for continuous state spaces: Stochastic message-passing with quantitative guarantees. *Journal of Machine Learning Research (JMLR)*, **14**, 2799–2835.

Nowozin, S., Lampert, C. H., *et al.* (2011). Structured learning and prediction in computer vision. *Foundations and Trends® in Computer Graphics and Vision*.

Owens, A., Xiao, J., Torralba, A., and Freeman, W. T. (2013). Shape anchors for data-driven multi-view reconstruction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Pacheco, J., Zuffi, S., Black, M. J., and Sudderth, E. (2014). Preserving modes and messages via diverse particle selection. In *Proc. of the International Conf. on Machine learning (ICML)*.

Peng, J., Hazan, T., McAllester, D., and Urtasun, R. (2011). Convex max-product algorithms for continuous mrfs with applications to protein folding. In *Proc. of the International Conf. on Machine learning (ICML)*.

Pollard, T. and Mundy, J. L. (2007). Change detection in a 3-d world. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Pollefeys, M. (2008). Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision (IJCV)*, **78**(2-3), 143–167.

Prisacariu, V. A. and Reid, I. (2011a). Nonlinear shape manifolds as shape priors in level set segmentation and tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Prisacariu, V. A. and Reid, I. (2011b). Shared shape spaces. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Prisacariu, V. A. and Reid, I. D. (2012). PWP3D: real-time segmentation and tracking of 3d objects. *International Journal of Computer Vision (IJCV)*, **98**(3), 335–354.

Ramachandran, V. S. (1986). Capture of stereopsis and apparent motion by illusory contours. *Perception & Psychophysics*, **39**(5), 361–373.

Ramachandran, V. S. and Rogers-Ramachandran, D. (2009). Two eyes, two views: Your brain and depth perception. *Scientific American Mind*.

Ramachandran, V. S., Madhusudhan Rao, V., and Vidyasagar, T. R. (1973). The role of contours in stereopsis. *Nature*, **242**, 412–414.

Ranftl, R., Pock, T., and Bischof, H. (2013). Minimizing TGV-based variational models with non-convex data terms. In *Proc. of the International Conf. on Scale Space and Variational Methods in Computer Vision (SSVM)*.

Ranftl, R., Bredies, K., and Pock, T. (2014). Non-local total generalized variation for optical flow estimation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Ranjan, A. and Black, M. J. (2016). Optical flow estimation using a spatial pyramid network. *arXiv.org*, **1611.00850**.

Ren, C. Y., Prisacariu, V., Murray, D., and Reid, I. (2013). Star3d: Simultaneous tracking and reconstruction of 3d objects using rgb-d data. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Revaud, J., Weinzaepfel, P., Harchaoui, Z., and Schmid, C. (2015). EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Rhemann, C., Hosni, A., Bleyer, M., Rother, C., and Gelautz, M. (2011). Fast cost-volume filtering for visual correspondence and beyond. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Richardt, C., Orr, D., Davies, I. P., Criminisi, A., and Dodgson, N. A. (2010). Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Rother, C., Kohli, P., Feng, W., and Jia, J. (2009). Minimizing sparse higher order energy functions of discrete variables. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H. J., and Davison, A. J. (2013). SLAM++: simultaneous localisation and mapping at the level of objects. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Saxena, A., Schulte, J., and Ng, A. Y. (2007). Depth estimation using monocular and stereo cues. In *Proc. of the International Joint Conf. on Artificial Intelligence (IJCAI)*.

Saxena, A., Sun, M., and Ng, A. Y. (2009). Make3D: learning 3D scene structure from a single still image. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **31**, 824–840.

Scaramuzza, D. and Siegwart, R. (2008). Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Trans. on Robotics*, **24**(5), 1015–1026.

Scharstein, D. and Pal, C. (2007). Learning conditional random fields for stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, **47**, 7–42.

Scharstein, D. and Szeliski, R. (2003). High-accuracy stereo depth maps using structured light. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nesic, N., Wang, X., and Westling, P. (2014). High-resolution stereo datasets with subpixel-accurate ground truth. In *Proc. of the German Conference on Pattern Recognition (GCPR)*.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, **26**(2), 214–226.

Schoenemann, T. and Cremers, D. (2008). High resolution motion layer decomposition using dual-space graph cuts. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Schönbein, M. and Geiger, A. (2014). Omnidirectional 3d reconstruction in augmented manhattan worlds. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*.

Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Seki, A. and Pollefeys, M. (2016). Patch based confidence prediction for dense disparity map. In *Proc. of the British Machine Vision Conf. (BMVC)*.

Seki, A. and Pollefeys, M. (2017). SGM-Nets: Semi-global matching with neural networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Sevilla-Lara, L., Sun, D., Jampani, V., and Black, M. J. (2016). Optical flow with semantic segmentation and localized layers. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Shum, H.-Y. and Szeliski, R. (2000). Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision (IJCV)*, **36**(2), 101–130.

Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., and Moreno-Noguer, F. (2015). Discriminative learning of deep convolutional feature point descriptors. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proc. of the International Conf. on Learning Representations (ICLR)*.

Sinha, S. N., Steedly, D., and Szeliski, R. (2009). Piecewise planar stereo for image-based rendering. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Song, S. and Xiao, J. (2014). Sliding shapes for 3D object detection in depth images. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Steinbrucker, F., Kerl, C., and Cremers, D. (2013). Large-scale multi-resolution surface reconstruction from rgb-d sequences. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Steinbruecker, F., Sturm, J., and Cremers, D. (2014). Volumetric 3d mapping in real-time on a cpu. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Sudderth, E. B., Ihler, A. T., Freeman, W. T., and Willsky, A. S. (2003). Nonparametric belief propagation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Sun, D., Sudderth, E. B., and Black, M. J. (2010). Layered image motion with explicit occlusions, temporal consistency, and depth ordering. In *Advances in Neural Information Processing Systems (NIPS)*.

Sun, D., Sudderth, E. B., and Black, M. J. (2012). Layered segmentation and optical flow estimation over time. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Sun, D., Wulff, J., Sudderth, E., Pfister, H., and Black, M. (2013). A fully-connected layered model of foreground and background flow. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Sun, D., Roth, S., and Black, M. J. (2014). A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision (IJCV)*, **106**(2), 115–137.

Szeliski, R. (2011). *Computer Vision - Algorithms and Applications*. Texts in Computer Science. Springer.

Taguchi, Y., Wilburn, B., and Zitnick, C. L. (2008). Stereo reconstruction with mixed pixels using adaptive over-segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Tola, E., Lepetit, V., and Fua, P. (2010). Daisy: An efficient dense descriptor applied to wide baseline stereo. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **32**(5), 815–830.

Trinh, H. and McAllester, D. (2009). Unsupervised learning of stereo vision with monocular cues. In *Proc. of the British Machine Vision Conf. (BMVC)*.

Tsai, A., III, W. M. W., Warfield, S. K., and Willsky, A. S. (2005). An em algorithm for shape classification based on level sets. *Medical Image Analysis (MIA)*, **9**(5), 491–502.

Uijlings, J. R., van de Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International Journal of Computer Vision (IJCV)*, **104**(2), 154–171.

Ulusoy, A. O., Geiger, A., and Black, M. J. (2015). Towards probabilistic volumetric reconstruction using ray potentials. In *Proc. of the International Conf. on 3D Vision (3DV)*.

Unger, C. (2013). *Contributions to Stereo Vision*. Dissertation, Technische Universität München.

Unger, M., Werlberger, M., Pock, T., and Bischof, H. (2012). Joint motion estimation and segmentation of complex scenes with label costs and occlusion modeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Vineet, V., Miksik, O., Lidegaard, M., Niessner, M., Golodetz, S., Prisacariu, V. A., Kahler, O., Murray, D. W., Izadi, S., Perez, P., and Torr, P. H. S. (2015). Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Vogel, C., Roth, S., and Schindler, K. (2013a). An evaluation of data costs for optical flow. In *Proc. of the German Conference on Pattern Recognition (GCPR)*.

Vogel, C., Schindler, K., and Roth, S. (2013b). Piecewise rigid scene flow. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Vogel, C., Roth, S., and Schindler, K. (2014). View-consistent 3D scene flow estimation over multiple frames. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

von Helmholtz, H. (1867). *Handbuch der physiologischen Optik*. L. Voss.

Wainwright, M. J. and Jordan, M. I. (2008). *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers.

Wang, S., Fidler, S., and Urtasun, R. (2015). Holistic 3d scene understanding from a single geo-tagged image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Wei, D., Liu, C., and Freeman, W. (2014). A data-driven regularization model for stereo and flow. In *Proc. of the International Conf. on 3D Vision (3DV)*.

Weinmann, M., Osep, A., Ruiters, R., and Klein, R. (2013). Multi-view normal field integration for 3D reconstruction of mirroring objects. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. (2013). DeepFlow: Large displacement optical flow with deep matching. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Wendel, A., Maurer, M., Graber, G., Pock, T., and Bischof, H. (2012). Dense reconstruction on-the-fly. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Woodford, O., Torr, P., Reid, I., and Fitzgibbon, A. (2009). Global stereo reconstruction under second-order smoothness priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **31**, 2115–2128.

Wulff, J. and Black, M. J. (2015). Efficient sparse-to-dense optical flow estimation using a learned basis and layers. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Xiao, J. and Furukawa, Y. (2014). Reconstructing the world's museums. *International Journal of Computer Vision (IJCV)*, **110**(3), 243–258.

Xu, J., Ranftl, R., and Koltun, V. (2017). Accurate optical flow via direct cost volume processing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Yamaguchi, K., Hazan, T., McAllester, D., and Urtasun, R. (2012). Continuous markov random fields for robust stereo estimation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Yamaguchi, K., McAllester, D., and Urtasun, R. (2013). Robust monocular epipolar flow estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Yamaguchi, K., McAllester, D., and Urtasun, R. (2014). Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Yang, J. and Li, H. (2015). Dense, accurate optical flow estimation with piecewise parametric model. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Yedidia, J. S., Freeman, W. T., and Weiss, Y. (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. on Information Theory*, **51**(7), 2282–2312.

Yoon, K.-J. and Kweon, I.-S. (2005). Locally adaptive support-weight approach for visual correspondence search. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Yu, F. and Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In *Proc. of the International Conf. on Learning Representations (ICLR)*.

Zach, C. (2008). Fast and high quality fusion of depth maps. In *Proc. of the International Conf. on 3D Digital Imaging, Modeling, Data Processing, Visualization and Transmission (THREEDIMPVT)*.

Zach, C., Pock, T., and Bischof, H. (2007a). A duality based approach for realtime TV-L1 optical flow. In *Proc. of the DAGM Symposium on Pattern Recognition (DAGM)*, pages 214–223.

Zach, C., Pock, T., and Bischof, H. (2007b). A globally optimal algorithm for robust tv-l1 range image integration. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Zagoruyko, S. and Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Zbontar, J. and LeCun, Y. (2014). Computing the stereo matching cost with a convolutional neural network. *arXiv.org*, **1409.4326**.

Žbontar, J. and LeCun, Y. (2016). Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research (JMLR)*, **17**(65), 1–32.

Zhang, C., Li, Z., Cai, R., Chao, H., and Rui, Y. (2014a). As-rigid-as-possible stereo under second order smoothness priors. In *Proc. of the European Conf. on Computer Vision (ECCV)*.

Zhang, G., Jia, J., Wong, T.-T., and Bao, H. (2009). Consistent depth maps recovery from a video sequence. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, **31**(6), 974–988.

Zhang, Q., Song, X., Shao, X., Zhao, H., and Shibasaki, R. (2014b). Start from minimum labeling: Learning of 3d object models and point labeling from a large and complex environment. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*.

Zhang, Q., Song, X., Shao, X., Zhao, H., and Shibasaki, R. (2014c). When 3d reconstruction meets ubiquitous RGB-D images. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Zhou, C., Güney, F., Wang, Y., and Geiger, A. (2015). Exploiting object similarity in 3d reconstruction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Zhou, Q.-Y., Miller, S., and Koltun, V. (2013). Elastic fragments for dense scene reconstruction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

Zimmer, H., Bruhn, A., and Weickert, J. (2011). Optic flow in harmony. *International Journal of Computer Vision (IJCV)*, **93**(3), 368–388.

Zitnick, C. L. and Dollár, P. (2014). Edge boxes: Locating object proposals from edges. In *Proc. of the European Conf. on Computer Vision (ECCV)*.