Semantic Three-Dimensional Understanding of Dynamic Scenes

by

Zhile Ren

B. S., Zhejiang University, 2013

Sc. M., Brown University, 2015

A dissertation submitted in partial fulfillment of the

requirements for the Degree of Doctor of Philosophy

in the Department of Computer Science at Brown University

Providence, Rhode Island

May 2018

This dissertation by Zhile Ren is accepted in its present form by
the Department of Computer Science as satisfying the dissertation requirement
for the degree of Doctor of Philosophy.

Date _____ _____
Erik B. Sudderth, Director

Recommended to the Graduate Council

Date _____ _____
Pedro F. Felzenszwalb, Reader

Date _____ _____
James Tompkin, Reader

Approved by the Graduate Council

Date _____ _____
Andrew G. Campbell
Dean of the Graduate School

Abstract of "Semantic Three-Dimensional Understanding of Dynamic Scenes" by Zhile Ren, Ph.D., Brown University, May 2018.

We develop new representations and algorithms for three-dimensional (3D) scene understanding from images and videos. To model cluttered indoor scenes, we introduce object descriptors that account for camera viewpoint, and use structured learning to perform 3D object detection and room layout prediction. We further boost accuracy by using latent support surfaces to capture style variations of objects and help detect small objects. In outdoor environments, we incorporate semantic segmentation in a cascaded prediction framework to more accurately model the 3D scene flow of moving objects.

We first propose a cloud of oriented gradient (COG) descriptor that links the 2D appearance and 3D pose of object categories, and thus accurately models how perspective projection affects perceived image boundaries. We also propose a Manhattan voxel representation which better captures room layout geometry. Effective classification rules are learned via a structured prediction framework. Contextual relationships among categories and layout are captured via a cascade of classifiers. Furthermore, we design algorithms that use latent support surfaces to better represent the 3D appearance of large objects, and provide contextual cues to improve the detection of small objects. Our model is learned solely from annotated RGB-D images, but nevertheless its performance substantially exceeds the state-of-the-art on the SUN RGB-D database.

We then focus on outdoor scene flow prediction. Many existing approaches use superpixels for regularization. We instead assume that scenes consist of foreground objects rigidly moving in front of a static background, and use semantic cues to produce pixel-accurate scene flow estimates. Our cascaded classification framework accurately models scenes by iteratively refining semantic segmentation masks, stereo correspondences, 3D rigid motion estimates, and optical flow fields. Our method has state-of-the-art performance on the challenging KITTI autonomous driving benchmark.

Vita

Zhile Ren was born on Feb. 11, 1991 in Beijing, China. He is a PhD student in Brown University working with Erik Sudderth since 2013. His research interests are computer vision with applications in 3D scene understanding and computer graphics with applications in image manipulation. During his PhD study he did research internships at Microsoft and NVIDIA. He did his undergrad in mathematics at Zhejiang University from 2009 to 2013.

Publications

- **Zhile Ren**, Erik Sudderth, 3D Object Detection with Latent Support Surfaces, IEEE Computer Vision and Pattern Recognition **(CVPR 2018)**
- **Zhile Ren**, Deqing Sun, Jan Kautz, Erik Sudderth, Cascaded Scene Flow Prediction using Semantic Segmentation, International Conference on 3D Vision **(3DV 2017)**
- **Zhile Ren**, Erik Sudderth, 3D Object Detection and Layout Prediction using Clouds of Oriented Gradients, IEEE Computer Vision and Pattern Recognition **(CVPR 2016)**
- Lingzhu Xiang, **Zhile Ren**, Mengrui Ni, Chad Jenkins, Robust Graph SLAM in Dynamic Environments with Moving Landmarks, International Conference on Intelligent Robots and Systems **(IROS 2015)**
- Pierre-Yves Laffont, **Zhile Ren**, Xiaofeng Tao, Chao Qian, James Hays, Transient Attributes for High-Level Understanding and Editing of Outdoor Scenes, ACM Transactions on Graphics **(SIGGRAPH 2014)**
- **Zhile Ren**, Greg Shakhnarovich, Image Segmentation by Cascaded Region Agglomeration, IEEE Computer Vision and Pattern Recognition **(CVPR 2013)**

Acknowledgements

My PhD journey has been joyful and fruitful thanks to many people! When I arrived in Brown, I was warmly welcomed by Erik Sudderth and his wonderful research group. Whenever I met Erik, his eyes were always filled with sparks, and he seemed to have perfect solutions to every scientific problem. Erik was very patient with me and he also showed me how to write structured, coherent and solid papers. Our research group was never short of imagination, rigorous scientific ideas, and love. Soumya Ghosh told me how to make the most of resources from the department. Jason Pacheco was like a human dictionary of machine learning, and he saved a lot of time for me to look up descriptions of algorithms. Mike Hughes organized a cool machine learning reading group, from which I learned many new things outside of my field. Dae Il Kim showed me how to give good presentations and we remain to be collaborators on research projects. Junior students Geng Ji, Gabe Hope, and Leah Weiner always gave valuable feedbacks to my papers and talks, and they were the ones who made sure my presentations were approachable to a broader audience. I wish the entire group can have gatherings whenever there is a chance.

My gratitude also goes to my awesome thesis committee members. James Tompkin was working in a neighboring office of mine, and his door was always open for me. Pedro Felzenszwalb taught an optimization class and organized an interesting pattern theory seminar. I learned many essential skills in those great events. Also, Stefanie Tellex served as a committee member for my first-year research comp, and she encouraged me to think about the connection between vision and robotics.

I would also like to thank other faculty members who shaped my mind. Greg Shakhnarovich from TTI-Chicago led me into computer vision research in my undergrad study, and I'm forever in debt of his support and encouragement. I met James Hays in CVPR shortly before my PhD study.

He invited me to his lab in my first year and we ended up working on a very interesting paper. During that time, Pierre-Yves Laffont served as my mentor and gave me many instructions in doing research.

During my PhD study, I have done three internships, and all my mentors broadened my imagination in computer vision. Sing Bing Kang gave me advice on how to interact with researchers in the community. Johannes Kopf demonstrated how to write clean code. Deqing Sun introduced various aspects of optical flow to me. Orazio Gallo helped me develop a well-organized research plan. Ming-Hsuan Yang and Jan Kautz gave me history lessons for AI and computer vision. With their support and help, my summers were filled with laughter and happy memories.

In Brown I was actively involved in Machine Learning Reading Group and Computer Vision Reading Group. Those were very interesting events! Besides Erik's students, I thank regular attendees DK Choe, Genevieve Patterson, Geoffrey Sun, Jeroen Chua, Sobhan Naderi Parizi. We became good friends in those weekly gatherings and I definitely learned so much from their talks.

And of course, I would like to thank amazing administrative and technical staffs in the department. I thank Lauren Clark for her prompt responses to all the administrative matters I encountered, and thanks John Bazik for maintaining computing grids for my research.

In my last year of PhD, I visited UC Irvine following Erik's career move. In this new place, I was welcomed by an awesome computer vision lab led by Charless Fowlkes. Charless always has a sharp mind identifying most crucial problems of computer vision projects. Along with his students Bailey Kong, Minhaeng Lee, Zhe Wang, Shu Kong, Phuc Nguyen, and Daeyun Shin, they gave me many feedbacks on my research. I hope we can continue collaborating in the future. I would also like to thank Qi Lou and Yu Guo for helping me settle down in Irvine and thanks Disi Ji for making sure my everyday meal plan was not boring in SoCal.

I would like to express special thanks to several great friends who helped me during my PhD study. Thanks my awesome office mate Rebecca Pankow, she showed me how to live a healthy life. I will keep her drawings and cool motivational penguins in my collection, and I hope we can root for each other as always. Thanks Yiming Li, he was the first fellow student I met in the department and we remain to be great friends. I wish one day my programming skill can match half of his talent. Thanks Songtao Chen & Xinyi Zhang and Dongkun Zhang & Xuan Zhang, they were the first few

people I met in Brown and I constantly felt their caring and kindness. Thanks Zhiqiang Sui & Da Yu for exploring New England with me. Thanks my trainer Corey and chiropractor Brandon, who literally shaped my body and kept me healthy and strong. I also thank the fantastic Astronomy Club led by Sherry. Along with Tracey and Hui, their creativity made my everyday life sparkling, and their comforting words made my PhD journey cheerful.

Last but not least, my parents have been extremely supportive to every single decision I made in my life. They always encourage me to try new things and find joy in life. I cannot ask more for their caring, and I hope to repay the favor in the future.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Scene understanding started with the goal of building machines that can see like humans to infer general principles and current situations from imagery, but it has become much broader than that [1]. In recent years, algorithmic advances have enabled many real-world vision applications, such as large scale scene classification [103, 155], object detection [43, 49], semantic scene parsing [66, 156] and data-driven computational photography [116, 68]. Nevertheless, those tasks remain notoriously difficult [1], especially the problem of 3D scene understanding.

3D scene understanding is one of the core tasks of computer vision, and seminal works on geometrical reconstruction and visual SLAM date back to decades ago [28, 105, 142]. With 3D understanding of scenes, intellegent agents such as robots or autonomous cars could better interact with and reason about objects in dynamic environments. However, acquiring 3D data and creating benchmarks has been challenging because 3D sensing technologies may be expensive or hard to use in general environments. In recent years rapid progress of range sensing technology, such as the *Microsoft Kinect* and *Velodyne LiDAR systems*, opened up exciting research on large-scale learning with RGB-D datasets [70, 40, 118, 117, 16, 2]. Training on those dataset, there has been progress on several challenging 3D understanding tasks such as object classification [123], point cloud segmentation [91], object proposal [19] and geometric reconstruction [122]. In this thesis, we focus on semantic 3D scene understanding of indoor and outdoor environments and propose new representations and learning frameworks to perform various 3D scene parsing tasks.

1.1 3D Indoor Scene Understanding

Indoor scene understanding is a core task for many exciting real-world applications such as robotics, virtual and augmented reality, and real-estate. The ultimate goal of holistic indoor scene understanding is to accurately answer questions like "Where is the chair?", "What is on the table?", "What is the layout of this room?", and so on. However, scene understanding in cluttered indoor scenes is very challenging because objects are occluding each other and have different shapes or styles. With depth sensors providing extra 3D knowledge of indoor scenes, we introduce novel algorithms for 3D indoor scene understanding. In particular, we focus on two of the most important tasks: object detection and room layout prediction.

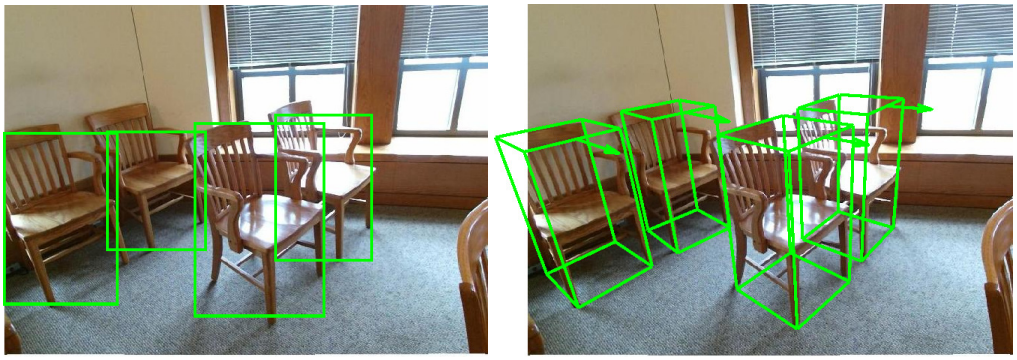


Figure 1.1: Comparing to traditional 2D bounding box annotations for indoor objects, 3D cuboid annotations contain richer information about object location, size, and orientation.

Describing objects by 2D bounding boxes has limited power in real-world vision and robotics applications. For example, when presented only with 2D bounding box detections for indoor objects (Fig. 1.1 left), robots are unable to perform complex environmental interactions like determining where to place objects, navigating, and figuring out contextual relationship among objects. Recently, 3D cuboids are increasingly used as a new way to represent objects in visual scenes. 3D cuboid annotations contain richer information about object orientations, physical sizes, and occupancy areas in the real world. The task of object detection using 3D cuboids has become a new standard in popular indoor and outdoor scene understanding benchmarks [118, 40].

Indoor objects in 3D environments are placed in various poses, and thus camera-captured objects in images vary widely in appearance. 3D objects also have diverse styles and shapes, and it is a

very challenging task to design reliable and accurate object descriptors. Moreover, due to sensor noise and the complexity of cluttered indoor environments, room layout prediction algorithms are very sensitive to outliers and thus usually inaccurate. In this thesis, we aim to solve those problems by introducing *cloud of oriented gradients* (COG), a view-invariant 3D descriptor for indoor object detection. Building on this, we design *Manhattan Voxels* to predict room layouts efficiently, and use *latent support surfaces* to encode object styles as well as help detect small objects. Finally, we design a *cascaded prediction* framework to reason about contextual relationship among objects, and achieve state-of-the-art performance on the SUN RGB-D dataset [118].

1.2 Motion Estimation in Outdoor Scenes

Given a video sequence of scenes with moving objects, the task of motion estimation aims to estimate the moving direction and magnitude of each pixel. This pixel-wise motion field is called *optical flow*. 2D motion estimation has a variety of applications in different areas of computer vision such as object tracking and image segmentation. However, there is a growing need to estimate accurate 3D motions, especially in real-world autonomous driving applications.

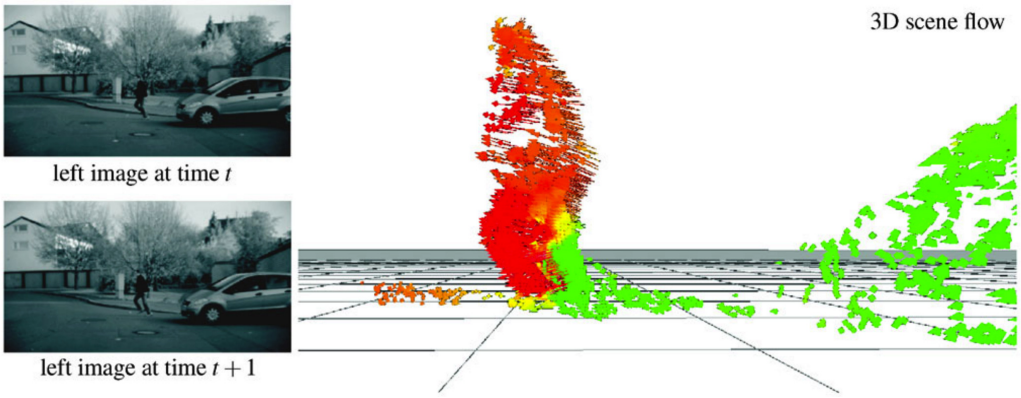


Figure 1.2: With stereo pairs of two consecutive frames as input, scene flow is the 3D motion for each pixel in the image [138].

The dense 3D motion of all points in an observed scene is called the *scene flow*. The task is challenging because scene flow models usually involve very large numbers of variables. Moreover, real-world scenes contain dynamic motions, and it is difficult to estimate per-pixel motion effectively

without understanding the semantics of the scene. In this thesis, we use semantic segmentation to separate foreground and background regions, and use a *cascaded prediction* approach to efficiently model segmentation, 3D geometry, and motion. Our *semantic scene flow* approach achieves comparable performance with state-of-the-art algorithms on the challenging KITTI autonomous driving benchmark [40].

1.3 Overview of Contributions

In this thesis, we design new representations and algorithms for 3D scene understanding from cluttered indoor RGB-D images and outdoor video sequences. We introduce novel representations for 3D object detection systems that localize objects with cuboids and describe room layouts by Manhattan structures. Using view-invariant 3D features, we capture 3D style variations and design systems to detect small objects by modeling support surfaces. Finally, we develop cascaded prediction frameworks to model 3D contextual relationships and enable rapid understanding of scene properties including depth, motion, and segmentation.

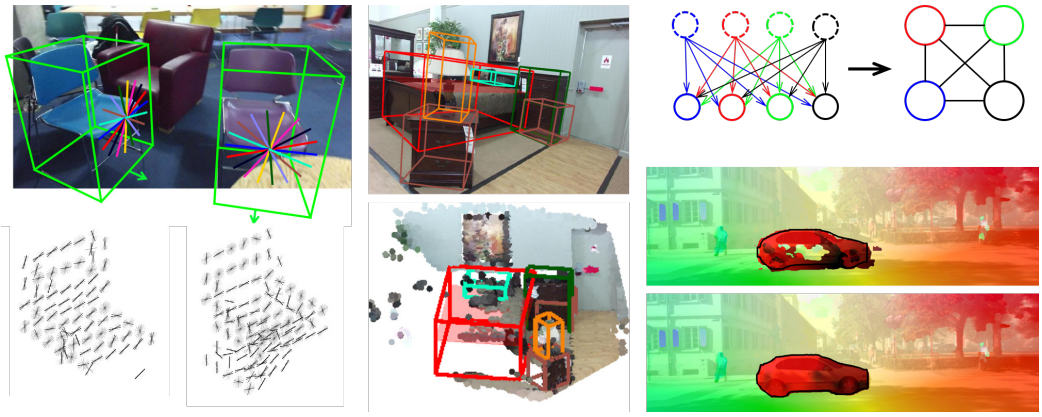


Figure 1.3: An overview of major contributions of this thesis. We introduce novel 3D descriptors for 3D object detection and use semantic segmentation to improve 3D scene flow estimation. Through this thesis we design cascaded prediction frameworks to model contextual relationship among objects and perform accurate and efficient inference on complex graph structures.

In Chapter 2, we introduce backgrounds of 3D visual scene understanding tasks related to this thesis, especially object detection, optical flow and stereo matching. We describe basic concepts

and survey related works for each computer vision task.

In Chapter 3, we introduce *cloud of oriented gradients* (COG), a novel object descriptor for 3D cuboids that are invariant to orientations. We use COG to design 3D object detection and room layout prediction algorithms. We also propose to use a *cascaded prediction* framework to model 3D contextual relationship among objects and room layouts. The two-stage holistic indoor scene understanding system is accurate and efficient, and has state-of-the-art performance on the large-scale SUN RGB-D benchmark [118].

We then propose *latent support surfaces* as a modeling technique to encode 3D object styles. The relative height of support surfaces on large objects is incorporated in 3D object detection systems as latent variables, and the whole system can be trained using latent structural SVM [148]. Predicted support surfaces on large objects also serve as the search space for small objects such as lamps, thus improving the precision of those object detectors.

In Chapter 4, we further explore cascaded prediction by modeling semantic segmentation, stereo matching and optical flow prediction in a joint *semantic scene flow* algorithm. We focus on autonomous driving applications [40] and demonstrate that a complex optimization problem can be solved in an iterative manner using cascaded prediction.

Finally in Chapter 5, we conclude with discussions on promising future research directions. We think that models trained on photo-realistic synthetic datasets have potentials to solve challenging scene understanding tasks.

Chapter 2

Background

In this chapter, we review the concepts of several computer vision tasks that are related to this thesis. We start by discussing methods for designing accurate and efficient 2D object detection systems on single RGB images. Following the discussion we describe challenges in modeling 3D data and the motivations behind designing 3D object detection systems. We then introduce optical flow, stereo matching, and their connections to 3D scene flow. We also describe the cascaded prediction framework, a powerful approach to model contextual relationship. In each subsequent section we summarize related works and discuss their connections to this thesis.

2.1 Object Detection

Object detection is a widely studied problem in computer vision. We introduce some basic concepts and highlight most related works in the rich literature.

Dalal and Triggs [27] introduced the *histogram of oriented gradient* (HOG) descriptor to model 2D object appearance using image gradients (Fig. 2.1). Input images are first split into rectangular cells, then image gradients are binned according to a given orientation histogram. With ground truth annotations for each object category, we can train classifiers for each individual object. In test time, object detectors will scan images in a sliding-windows approach and report a confidence score for each bounding box proposal.

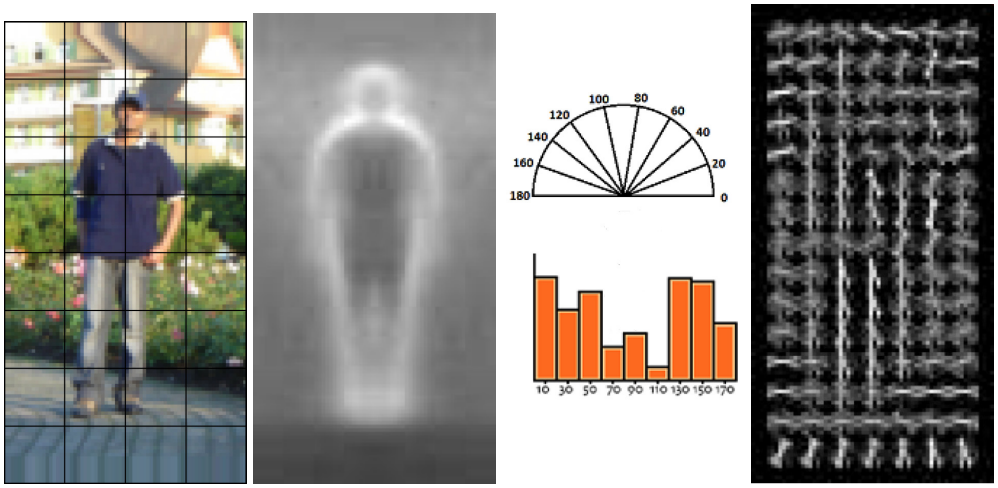


Figure 2.1: Visualizing HOG descriptors [27]. Image gradients are binned in orientation histograms in each cell. The concatenated features in each cell act as appearance descriptors for pedestrians.

A bounding box proposal B_p is considered as a correct prediction when the intersection-over-union [32] score (IOU) with ground truth B_{gt} exceed 50%.

$$\text{IOU} = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})} \tag{2.1}$$

To evaluate the performance of an object detector, a precision-recall curve is computed from ranked scores, and average precision (AP) score determines the accuracy of the algorithm.

Building on HOG, Felzenszwalb *et al*. [33] used a discriminately-trained part-based model to represent objects. This method is effective because it explicitly models object parts as latent variables and thus implicitly encode object style variations. More recently, many papers have used convolutional neural networks (CNNs) to extract rich features from images [43, 42, 97, 49, 78]. We include a nice visualization by Girshick *et al*.in Fig. 2.2 to demonstrate the general approach. The hand-crafted HOG [27] feature is replaced with outputs from CNNs, leading to state-of-the-art performance with efficient detection speed [95, 96].

Although accuracies of object detectors have improved rapidly in recent years, in cluttered indoor scenes, localizing objects in 2D remains a challenging task. This is because indoor objects are heavily occluded and have view/shape variations (Fig. 2.3). Those findings motivate us to

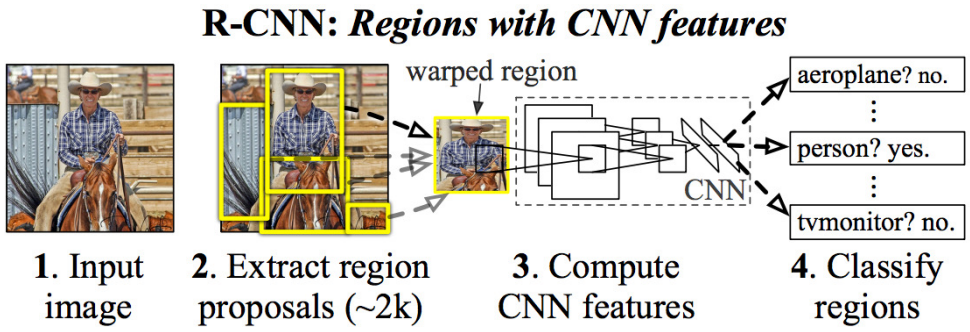


Figure 2.2: Visualization of a standard object detection system using CNNs [43].

propose new representations to describe objects, and use advanced depth sensors to solve scene understanding tasks in 3D.

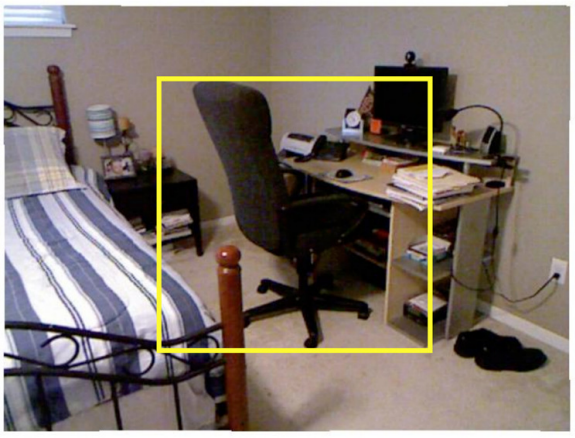


Figure 2.3: In cluttered indoor scenes, a chair is mis-classified as "Potter's wheel" using a popular object detector [113].

2.2 3D scene understanding

Increasingly, real-world computer vision systems often incorporate depth data as an additional input to increase accuracy and robustness (Fig. 2.4). With depth maps we can reconstruct point cloud representation of scenes. This 2.5D input is very helpful and is widely used for many indoor and outdoor scene understanding tasks. There have recently been significant advances in 3D object classification [141, 123], point cloud segmentation [91, 92], room layout prediction [71, 111], 3D

object context [115, 152], and 3D shape reconstruction [128, 25]. Here, we focus on the related problems of 3D object detection and 3D scene flow estimation.

Figure 2.4: Common depth sensors for indoor (top) and outdoor autonomous driving applications (bottom)

2.2.1 3D Object Detection

In outdoor scenes, localizing objects with 3D cuboids has become a new standard in the popular KITTI autonomous driving benchmark [40]. 3D detection systems model car shape and occlusion patterns using LiDAR or stereo inputs [20, 85, 143, 90], and may also incorporate additional bird's eye view data [21]. 3D cuboid representations are more powerful than 2D bounding boxes because they contain more information for object locations, physical occupancy spaces and orientation. However, most outdoor 3D detection systems are designed to identify vehicles and pedestrians, and those methods may not generalize well to more challenging tasks in cluttered indoor scenes.

In indoor scenes, objects have greater shape and style variations. Because indoor objects are often heavily occluded by their cluttered environments, localizing objects with 3D cuboids [77, 46] instead of 2D bounding boxes could be more effective. Some works aligned 3D CAD models to objects in RGB-D inputs [47, 119] and evaluated on the small-scale NYU Depth dataset [117], but the computational costs are usually expensive. A simple 3D convolutional neural network was designed to detect simple objects in real time [82]. In 2015, Song *et al.*introduced a larger scale SUN RGB-D dataset [118] that contain 10335 RGB-D images with accurate 3D cuboid annotations for indoor objects, room layouts and scene categories (Fig. 2.5). The size of the dataset matches that of PASCAL-VOC dataset [32] and motivates a line of research projects.

In recent years there were works that utilized pre-trained 2D detectors and region proposals as

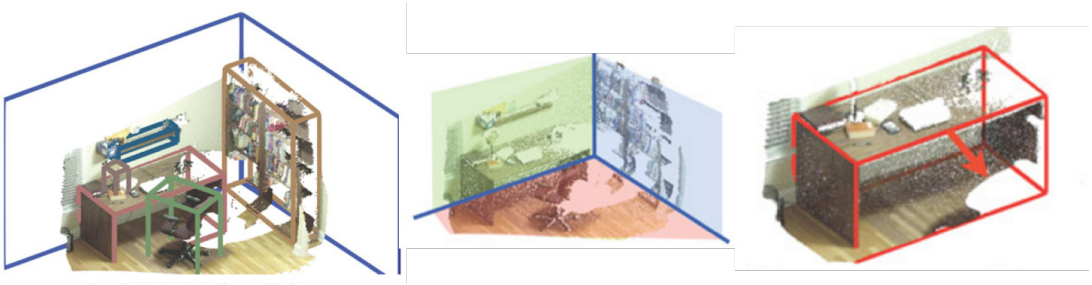


Figure 2.5: Visualizing 3D cuboid annotations for objects and room layouts in SUN RGB-D dataset [118].

priors [103], and localized 3D bounding boxes via a separate CNN [120, 29, 69]. Those methods can achieve good performance in real time, but are very dependent to the accuracy of 2D object detection systems. Without explicitly modeling view-dependent information and object style information, the accuracy of object detectors still has room for improvement. In this thesis we introduce the *clouds of oriented gradient* (COG) to represent 3D cuboids in orientation-invariant features, and we design a *latent support surface model* to encode object style variations.

Apart from detecting objects, we also focus on the task of 3D room layout estimation. The output of room layout prediction contains 3D locations of walls, ceiling and floor. There are works designed to predict such room structure in 2D [111, 151, 110, 50, 71, 81]. However, even if two layout predictions are similar in 2D, the reconstructed 3D layout may be completely different (Fig. 2.6). Recently there are more works that directly predict room layouts in 3D [157] using CNNs, and in this thesis we introduce an effective *Manhattan voxel* approach that predict 3D room layouts.

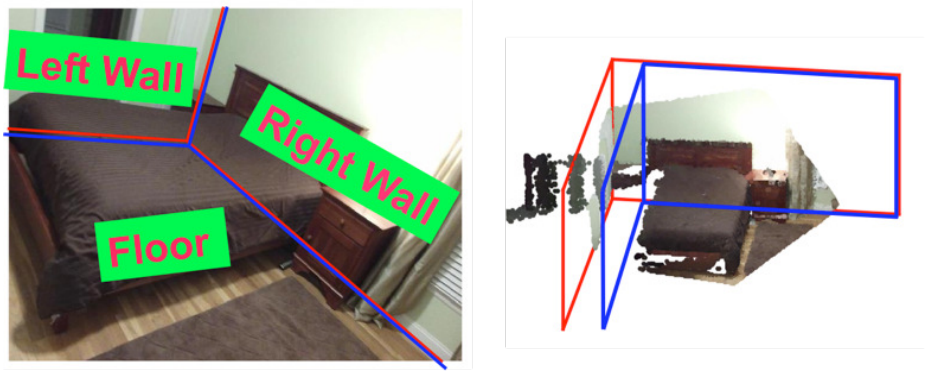


Figure 2.6: Similar layout predictions in 2D (in red and blue) may be completely different 3D.

Detecting support surfaces is an essential first step in understanding the geometry of 3D scenes for such tasks as surface normal estimation [137, 37] and shape retrieval [9]. Silberman *et al*. [117] use semantic segmentation to model object support relationships; this work was later extended by Guo *et al*. [45] for support surface prediction. However, support surfaces have not been previously used to enable 3D object detection. In this thesis, we treat support surfaces as latent variables to capture object style variations, and use them to localize small objects.

2.2.2 3D Scene Flow

Optical flow is also a widely studied computer vision task. Given two consecutive frames from input videos, optical flow is the 2D motion of every pixel caused by the movement of object or camera. It is usually visualized by color-coded maps that correspond to different orientations and magnitudes (Fig. 2.7). Since the seminal work of Horn and Schunck [54], a rich literature of variational approaches has flourished (e.g., [125, 13, 102]). Given the large number of publications in this area, we refer the reader to [125, 7] for a more complete survey, particularly of the two-frame based methods. The recent success of deep learning inspired several approaches based on convolutional neural networks. Dosovitskiy *et al*. [30] introduced a denoising autoencoder network, called FlowNet, for estimating optical flow. Mayer *et al*. [83] extended the FlowNet to disparity and scene flow estimation with a large synthetic dataset. Other related networks can either replace parts of the traditional pipeline [145, 6], or can be used in an end-to-end fashion [58, 94, 126]. While CNN models generate scene flow predictions rapidly, most networks trained on synthetic data are not competitive with state-of-the-art methods on the real-world KITTI benchmark [40, 84].

Similar to optical flow prediction, *stereo matching* is also a classical task for finding pixel-wise correspondences between images. Given a rectified pair of images from left and right cameras, we aim to find corresponding pixel pairs along each horizontal line. The predicted horizontal shift of each pixel is called *disparity*, and it can be used to recover the depth map. Both variational approaches [44, 112] and CNN-based methods [83, 79] have been proposed and we have seen many progress on public benchmarks [40].

With stereo inputs from consecutive video frames, Vedula *et al*. [130] first defined the *scene*

Figure 2.7: Visualizing inputs to optical flow prediction tasks and the ground truth motion map in MPI Sintel dataset [15].

flow as the dense 3D motion of all points in an observed scene, and recovered voxel-based flow estimates using 2D optical flow fields from several calibrated cameras. The 3D motion for each pixel in input frames can be predicted, giving autonomous systems more power to understand dynamic visual scenes. Huguet and Devernay [56] then proposed a variational approach and jointly solved for stereo and optical flow, while Wedel *et al*. [139] decoupled the stereo and flow problems for efficiency. These classic algorithms only improve marginally over modern, state-of-the-art stereo and optical flow methods.

Although scene flow algorithms require more input images than standard optical flow or stereo reconstruction methods, the task is still challenging due to the high dimensionality of the disparity and motion fields. To reduce the solution space, Vogel *et al*. [133] introduced a *piecewise rigid scene flow* (PRSF) model and used superpixels to constrain scene flow estimation. For the first time, they showed that scene flow methods could outperform stereo and optical flow methods by a large margin on the challenging KITTI dataset [40]. In a follow-up work they extended their formulation to multiple frames and improved accuracy [134]. However, because the PRSF model relies on bottom-up cues for superpixel segmentation, it tends to over-segment foreground objects such as cars. Over-segmented parts are allocated independent motion models, so global information cannot be effectively shared.

Inspired by the success of Vogel *et al*. [133], Menze and Geiger [84] annotated a new KITTI dataset with dynamic foreground objects for scene flow evaluation. They proposed an *object scene flow* (OSF) algorithm that segments the scene into independently moving regions, and encourages the superpixels within each region to have similar 3D motion. Although the performance of OSF improved on baselines, the "objects" in their model are assumed to be planar and initialized via bottom-up motion estimation, so physical objects are often over-segmented. The inference time required for the OSF method is also significantly longer than most competing methods.

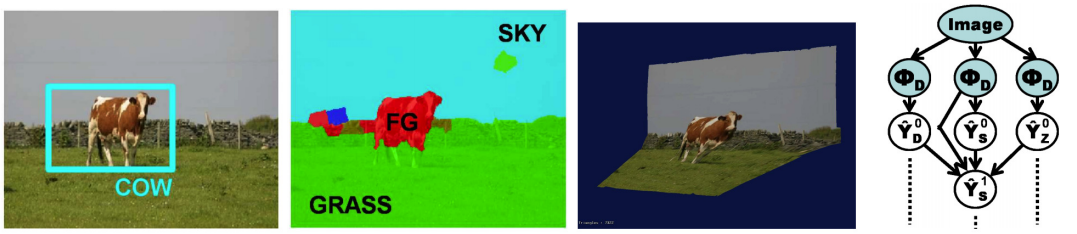
Some related work integrates automatic motion segmentation with optical flow prediction [89, 127], but assumes large differences between the motion of objects and cameras, and requires multiple input frames. Exploiting the recent success of CNNs for semantic segmentation [26, 154], semantic cues have been shown to improve optical flow estimation [5, 57, 114]. Concurrent work [10] also shows that semantic cues can improve scene flow estimation. In this thesis, we propose a coherent model of semantic segmentation, scene geometry, and object motion. We use a cascaded prediction framework [52] to efficiently solve this high-dimensional inference task. We evaluate our algorithm on the challenging KITTI dataset [84] and show that using semantic cues leads to state-of-the-art scene flow estimates.

2.3 Cascaded Prediction

Holistic scene understanding is a task that consists of semantic object reasoning, spatial contextual modeling, as well as scene type estimation [147, 118]. Traditionally, learning and inference in each sub-task is performed independently and integrated in conditional random fields (CRFs) [67]. For complex graph structures with high-dimensional output spaces, inference can be inefficient.

Cascaded Prediction is a nice framework that combines multiple instantiations of each classifier and organizes them into stages [52]. In graphical model representations, information in early stages are treated as latent variables, and they are used to infer variable values in future stages (See Fig. 2.8). By marginalizing latent variables from early stages, we can recover the fully connected undirectional graph structures as in CRFs, but the inference is much more efficient. Cascaded

prediction is widely used in many computer vision tasks such as face detection [131, 74], image segmentation [98, 26]. In this thesis, we apply cascaded prediction to perform holistic indoor scene understanding, and demonstrate its effectiveness in solving inference problems with high-dimensional outputs for outdoor scene flow estimation.



(a) Detected Objects (b) Classified Regions (c) 3D Structure (d) **CCM** Framework

Figure 2.8: Heitz *et al*. [52] estimate object detection, semantic segmentation and 3D structure in a cascaded prediction framework.

Chapter 3

Indoor RGB-D Scene Understanding: Object Detection and Layout Estimation

We develop new representations and algorithms for three-dimensional (3D) object detection and spatial layout prediction in cluttered indoor scenes. RGB-D images are traditionally described by local geometric features of the 3D point cloud. We propose a cloud of oriented gradient (COG) descriptor that links the 2D appearance and 3D pose of object categories, and thus accurately models how perspective projection affects perceived image boundaries. We also propose a "Manhattan voxel" representation which better captures the 3D room layout geometry of common indoor environments. Effective classification rules are learned via a structured prediction framework that accounts for the intersection-over-union overlap of hypothesized 3D cuboids with human annotations, as well as orientation estimation errors. Contextual relationships among categories and layout are captured via a cascade of classifiers, leading to holistic scene hypotheses with improved accuracy. Our model is learned solely from annotated RGB-D images, without the benefit of CAD models, but nevertheless its performance substantially exceeds the state-of-the-art on the SUN RGB-D database. Avoiding CAD models allows easier learning of detectors for many object categories.

Existing 3D representations for RGB-D images capture the local shape and appearance of object categories, but have limited power to represent objects with different visual styles. The detection of small objects is also challenging because the search space is very large in 3D scenes. However, we

observe that much of the shape variation within 3D object categories can be explained by the location of a latent support surface, and smaller objects are often supported by larger objects. Therefore, we explicitly use latent support surfaces to better represent the 3D appearance of large objects, and provide contextual cues to improve the detection of small objects. We evaluate our model with 19 object categories from the SUN RGB-D database, and demonstrate state-of-the-art performance.

3.1 Introduction

The last decade has seen major advances in algorithms for the semantic understanding of 2D images [32, 103]. Images of indoor (home or office) environments, which are typically highly cluttered and have substantial occlusion, are particularly challenging for existing models. Recent advances in depth sensor technology have greatly reduced the ambiguities present in standard RGB images, enabling breakthroughs in scene layout prediction [72, 51, 151], support surface prediction [117, 37, 45], semantic parsing [48], and object detection [119]. A growing number of annotated RGB-D datasets have been constructed to train and evaluate indoor scene understanding methods [104, 70, 117, 118].

A wide range of semantic 3D scene models have been developed, including approaches based on low-level voxel representations [63]. Generalizing the bounding boxes widely used for 2D detection, the 3D size, position, and orientation of object instances can be described by bounding *cuboids* (convex polyhedra). Several methods fit cuboid models to RGB or RGB-D data [60, 59, 144] but do not have any semantic, high-level scene understanding. Other work has used CRFs to classify cuboids detected by bottom-up grouping [77], or directly detected objects in 3D by matching to known CAD models in "sliding" locations [119].

Several recent papers have used CAD models as additional information for indoor scene understanding, by learning models of object shape [141] or hallucinating alternative viewpoints for appearance-based matching [3, 76, 75]. While 3D models are a potentially powerful information source, there does not exist an abundant supply of models for all categories, and thus these methods

have typically focused on a small number of categories (often, just chairs [3]). Moreover, example-based methods [119] may be computationally inefficient due to the need to match each examplar to each test image. It is unclear how many CAD models are needed to faithfully capture an object class.

To model the spatial layout of indoor scenes, many methods assume an orthogonal "Manhattan" structure [24] and aim to infer 2D projections of the 3D structure. Building on [72] and [53], Hedau et al. [50] use a structured model to rerank layout hypotheses, Schwing et al. [110] propose an efficient integral representation to efficiently explore exponentially many layout proposals, and Zhang et al. [151] incorporate depth cues. Jointly modeling objects may improve layout prediction accuracy [51, 111], but previous work has focused on restricted environments (e.g., beds that are nearly always aligned with walls) and may not generalize to more cluttered scenes. Other work has used point cloud data to directly predict 3D layout [77, 118], but can be sensitive to errors in RGB-D depth estimates.

Simple scene parsing algorithms detect each category independently, which can introduce many false positives even after non-maximum suppression. Previous work has used fairly elaborate, manually engineered heuristics to prune false detections [119] or used CAD models and layout cues jointly to model scenes [41]. In this paper we show that a *cascaded classification framework* [52] can be used to learn contextual relationships among object categories and the overall room layout, so that visually distinctive objects lead to holistic scene interpretations of higher quality.

However, existing 3D detection algorithms suffer some common problems. Given diverse objects in the same category, modeling different visual styles is often very challenging [35], and ground truth annotations of 3D cuboids can vary among different human annotators (see Fig. 3.9). Moreover, objects with smaller physical size are hard to detect because the search space in the whole scene is very big, and bottom-up proposals typically contain many false positives.

State-of-the-art 3D object features, such as TSDF [120], are calculated for a grid of voxels within each hypothesized 3D cuboid. A major cause of feature inconsistency across different object instances is variation in the location of the supporting surface contained by many indoor objects. We treat the height of the support surface as a latent variable, and use it to distinguish different visual styles of the same object category.

Modeling support surface can also help detect smaller objects like monitors, lamps, TVs, and pillows. Since small objects are typically placed on the supporting surfaces of large objects, we first detect large objects on the ground and predict their support surface location, and then search for small objects on top of support surface areas. The reduced search space for small objects naturally reduces false positives and improves performance.

In this chapter, we propose a general framework for learning detectors for multiple object categories using only RGB-D annotations (Sec. 3.2). We introduce a novel *cloud of oriented gradients* (COG) feature that robustly links 3D object pose to 2D image boundaries. We also introduce a new *Manhattan voxel* representation of 3D room layout geometry. We then use a structured prediction framework to learn an algorithm that aligns 3D cuboid hypotheses to RGB-D data, and a cascaded classifier to incorporate contextual cues from other object instances and categories, as well as the overall 3D layout.

Building on our proposed cascaded 3D scene understanding framework, another contributions in this chapter include the introduction of new 3D view features that improve 3D detection systems, the modeling of support surfaces as latent variables capturing intra-class variation for large objects, and the use of support surfaces to more accurately detect small objects (Sec. 3.3). We evaluate our algorithm on the SUN RGB-D dataset [118] and achieve state-of-the-art accuracy in the 3D detection of 19 object categories.

3.2 3D Cuboid Detection and Layout Prediction Using Cloud of Oriented Gradients

3.2.1 Modeling 3D Geometry & Appearance

Our object detectors are learned from 3D oriented cuboid annotations in the SUN-RGBD dataset [118], which contains 10,335 RGB-D images and 19 labeled object categories. We discretize each cuboid into a $6 \times 6 \times 6$ grid of (large) voxels, and extract features for these $6^3 = 216$ cells. Voxel dimensions are scaled to match the size of each instance. We use standard descriptors for the 3D geometry of the observed depth image, and propose a novel *cloud of oriented gradient* (COG) descriptor of RGB

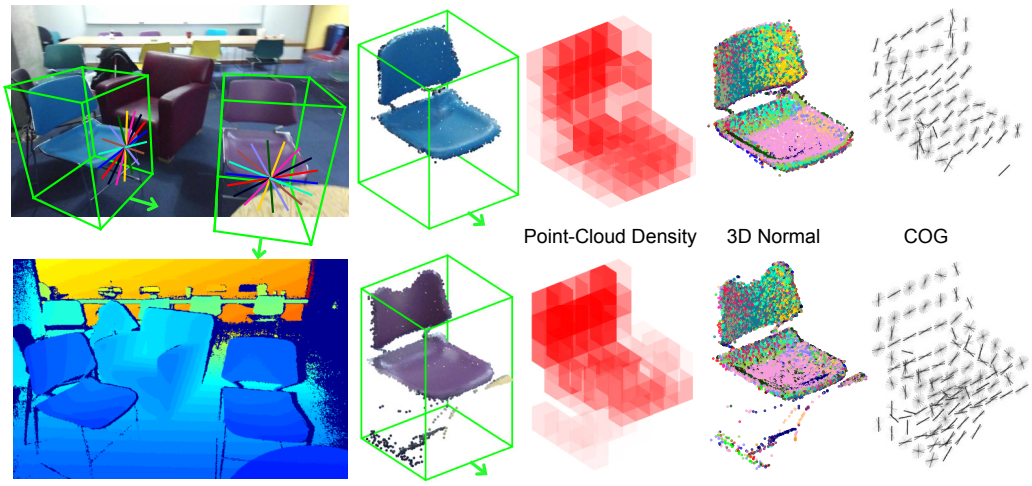


Figure 3.1: Given input RGB and Depth images (left), we align oriented cuboids and transform observed data into a canonical coordinate frame. For each voxel in a $6 \times 6 \times 6$ grid, we then extract (from left to right) point cloud density features, 3D normal orientation histograms, and our COG model of back-projected image gradient orientations. On the left, COG bins are colored to show alignment between instances. The value of the point cloud density feature is proportional to the voxel intensity, each 3D orientation histogram bin is assigned a distinct color, and COG feature intensities are proportional to the normalized energy in each orientation bin, similarly to HOG descriptors [27].

appearance. We also propose a *Manhattan voxel* model of 3D room layout geometry.

Object Geometry: 3D Density and Orientation

Point Cloud Density Conditioned on a 3D cuboid annotation or detection hypothesis i, suppose voxel ℓ contains $N_{i\ell}$ points. We use perspective projection to find the silhouette of each voxel in the image, and compute the area $A_{i\ell}$ of that convex region. The *point cloud density* feature for voxel ℓ then equals $\phi^a_{i\ell} = N_{i\ell}/A_{i\ell}$. Normalization gives robustness to depth variation of the object in the scene. We normalize by the local voxel area, rather than by the total number of points in the cuboid as in some related work [119], to give greater robustness to partial object occlusions.

3D Normal Orientations Various representations, such as spin images [62], have been proposed for the vectors normal to a 3D surface. As in [119], we build a 25-bin histogram of normal orientations within each voxel, and estimate the normal orientation for each 3D point via a plane fit to its 15 nearest neighbors. This feature ϕ^b_i captures the surface shape of cuboid i via patterns of local 3D orientations.

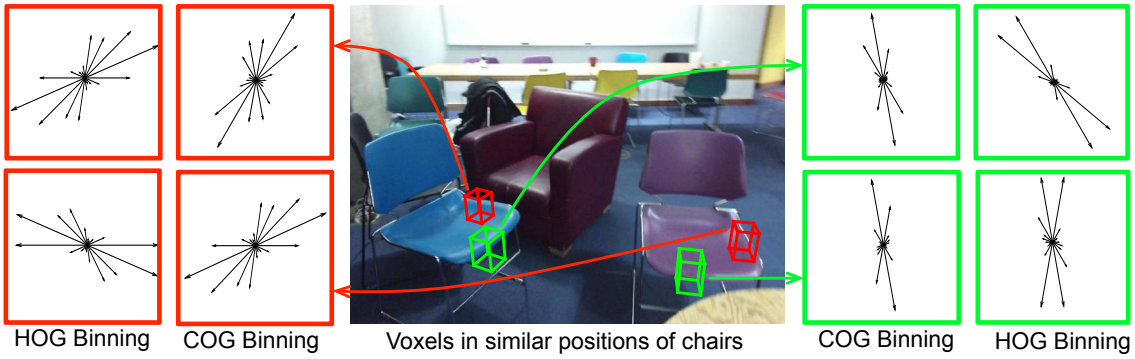


Figure 3.2: For two corresponding voxels (red and green) on two chairs, we illustrate the orientation histograms that would be computed by a standard HOG descriptor [27] in 2D image coordinates, and our COG descriptor in which perspective geometry is used to align descriptor bins. Even though these object instances are very similar, their 3D pose leads to wildly different HOG descriptors.

Clouds of Oriented Gradients (COG)

The *histogram of oriented gradient* (HOG) descriptor [27] forms the basis for many effective object detection methods [32]. Edges are a very natural foundation for indoor scene understanding, due to the strong occluding contours generated by common objects. However, gradient orientations are of course determined by 3D object orientation and perspective projection, so HOG descriptors that are naively extracted in 2D image coordinates generalize poorly.

To address this issue, some previous work has used 3D CAD models to hallucinate the edges that would be expected from various synthetic viewpoints [75, 3]. Other work has restrictively assumed that parts of objects are near-planar so that image warping may be used for alignment [35], or that all objects have a 3D pose aligned with the global "Manhattan world coordinates" of the room [51]. Some previous 3D extensions of the HOG descriptor [14, 109] assume that either a full 3D model or mesh model is given. In recent independent research [121], 3D cuboid hypotheses were used to aggregate standard 2D features from a deep convolutional neural network, but the relationship between these features and 3D object orientation was not modeled. Our *cloud of oriented gradient* (COG) feature accurately describes the 3D appearance of objects with complex 3D geometry, as captured by RGBD cameras in any orientation.

Gradient Computation We compute gradients by applying filters $[-1, 0, 1]$, $[-1, 0, 1]^T$ to the RGB channels of the unsmoothed 2D image. The maximum responses across color channels are the gradients (dx, dy) in the x and y directions, with corresponding magnitude $\sqrt{dx^2 + dy^2}$.

3D Orientation Bins The standard HOG descriptor [27] uses evenly spaced gradient bins, with 0° being the horizontal image direction. As shown in Fig. 3.2, this can produce very inconsistent descriptors for objects in distinct poses.

For each cuboid we construct nine 3D orientation bins that are evenly spaced from $0^\circ - 180^\circ$ in the half-disk sitting vertically along its horizontal axis. We then use perspective projection to find corresponding 2D bin boundaries. For each point that lies within a given 3D voxel, we accumulate its unsigned 2D gradient in the corresponding projected 2D orientation bin. To avoid image processing operations that can be unstable for objects with non-planar geometry, we accumulate standard gradients with warped histogram bins, rather than warping images to match fixed orientation bins.

Normalization and Aliasing We bilinearly interpolate gradient magnitudes between neighboring orientation bins [27]. To normalize the histogram $\phi^c_{i\ell}$ for voxel ℓ in cuboid i, we then set $\phi^c_{i\ell} \leftarrow \phi^c_{i\ell} / \sqrt{||\phi^c_{i\ell}||^2 + \epsilon}$ for a small $\epsilon > 0$. Accounting for all orientations and voxels, the dimension of the COG feature is $6^3 \times 9 = 1944$.

Room Layout Geometry: Manhattan Voxels

Given an RGB-D image, scene parsing requires not only object detection, but also room layout (floor, ceiling, wall) prediction [50, 72, 151, 111]. Such "free space" understanding is crucial for applications like robot navigation. Many previous methods treat room layout prediction as a 2D labeling task [4, 110, 151], but small mistakes in 2D can lead to huge errors in 3D layout prediction. Simple RGB-D layout prediction methods [118] work by fitting planes to the observed point cloud data. We propose a more accurate learning-based approach to predicting Manhattan geometries.

The orthogonal walls of a standard room can be represented via a cuboid [88], and we could define geometric features via a standard voxel discretization (Fig. 3.3, bottom left). However, because corner voxels usually contain the intersection of two walls, they then mix 3D normal vectors with

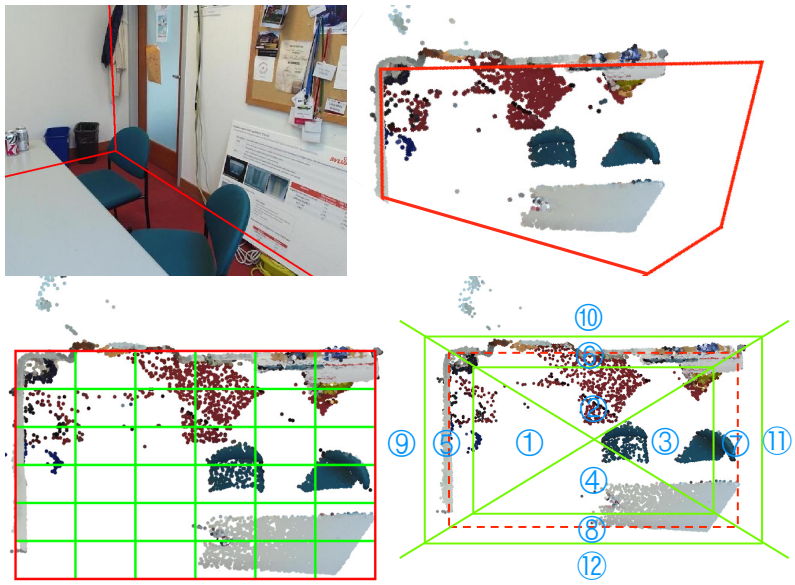


Figure 3.3: Models for 3D layout geometry. *Top:* Ground truth annotation. *Bottom:* Top-down view of the scene and two voxel-based quantizations. We compare a regular voxel grid (left) to our Manhattan voxels (right; dashed red line is the layout hypothesis).

very different orientations. In addition, this discretization ignores points outside of the hypothesized cuboid, and may match subsets of a room that have wall-like structure.

We propose a novel *Manhattan voxel* (Fig. 3.3, bottom right) discretization for 3D layout prediction. We first discretize the vertical space between floor and ceiling into 6 equal bins. We then use a threshold of $0.15m$ to separate points near the walls from those in the interior or exterior of the hypothesized layout. Further using diagonal lines to split bins at the room corners, the overall space is discretized in $12 \times 6 = 72$ bins. For each vertical layer, regions $R_{1:4}$ model the scene interior whose point cloud distribution varies widely across images. Regions $R_{5:8}$ model points near the assumed Manhattan wall structure: R_5 and R_6 should contain orthogonal planes, while R_5 and R_7 should contain parallel planes. Regions $R_{9:12}$ capture points outside of the predicted layout, as might be produced by depth sensor errors on transparent surfaces.

3.2.2 Learning to Detect Cuboids & Layouts

For each voxel ℓ in some cuboid B_i annotated in training image I_i, we have one point cloud density feature $\phi^a_{i\ell}$, 25 surface normal histogram features $\phi^b_{i\ell}$, and 9 COG appearance features $\phi^c_{i\ell}$. Our overall feature-based representation of cuboid i is then $\phi(I_i, B_i) = \{\phi^a_{i\ell}, \phi^b_{i\ell}, \phi^c_{i\ell}\}_{\ell=1}^{216}$. Cuboids are aligned via annotated orientations as illustrated in Fig. 3.1, using the gravity direction provided in the SUN-RGBD dataset [118]. Similarly, for each of the Manhattan voxels ℓ in layout hypothesis M_i we compute point cloud density and surface normal features, and $\phi(I_i, M_i) = \{\phi^a_{i\ell}, \phi^b_{i\ell}\}_{\ell=1}^{72}$.

Structured Prediction of Object Cuboids

For each object category c independently, using those images which contain visible instances of that category, our goal is to learn a prediction function $h_c : I \rightarrow B$ that maps an RGB-D image I to a 3D bounding box $B = (L, \theta, S)$. Here, L is the center of the cuboid in 3D, θ is the cuboid orientation, and S is the physical size of the cuboid along the three axes determined by its orientation. We assume objects have a base upon which they are usually supported, and thus θ is a scalar rotation with respect to the ground plane.

Given n training examples of category c, we use an n-slack formulation of the structural support vector machine (SVM) objective [61] with margin rescaling constraints:

$$\begin{aligned} \min_{w_c, \xi \geq 0} \quad & \frac{1}{2} w_c^T w_c + \frac{C}{n} \sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & w_c^T [\phi(I_i, B_i) - \phi(I_i, \bar{B}_i)] \geq \Delta(B_i, \bar{B}_i) - \xi_i, \quad \text{for all } \bar{B}_i \in \mathcal{B}_i, i = 1, \ldots, n. \end{aligned} \tag{3.1}$$

Here, $\phi(I_i, B_i)$ are the features for oriented cuboid hypothesis B_i given RGB-D image I_i, B_i is the ground-truth annotated bounding box, and $\mathcal{B}_i$ is the set of possible alternative bounding boxes. For training images with multiple instances, as in previous work on 2D detection [129] we add images multiple times to the training set, each time removing the subset of 3D points contained in other instances.

Given some ground truth cuboid B and estimated cuboid $\bar{B}$, we define the following loss function:

$$\Delta(B, \bar{B}) = 1 - \text{IOU}(B, \bar{B}) \cdot \left(\frac{1 + \cos(\bar{\theta} - \theta)}{2} \right). \tag{3.2}$$

Here, $\text{IOU}(B, \bar{B})$ is the volume of the 3D intersection of the cuboids, divided by the volume of their 3D union. The loss is bounded between 0 and 1, and is smallest when the $\text{IOU}(B, \bar{B})$ is near 1 *and* the orientation error $\theta - \bar{\theta} \approx 0$. Loss approaches 1 if either position or orientation is wrong.

We solve the loss-sensitive objective of Eq. (3.1) using a cutting-plane method [61]. We also experimented with detectors based on a standard binary SVM with hard negative mining, but found that the loss-sensitive S-SVM classifier is more accurate (see Fig. 3.5) and also more efficient in handling the large number of negative cuboid hypotheses.

Cuboid Hypotheses We precompute features for candidate cuboids in a sliding-window fashion using discretized 3D world coordinates, with 16 candidate orientations. We discretize cuboid size using empirical statistics of the training bounding boxes: $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ width quantiles, $\{0.25, 0.5, 0.75\}$ depth quantiles, and $\{0.3, 0.5, 0.8\}$ height quantiles. Every combination of voxel size, and 3D location and orientation, is then evaluated.

Structured Prediction of Manhattan Layouts

We again use the S-SVM formulation of Eq. (3.1) to predict Manhattan layout cuboids $M = (L, \theta, S)$. The loss function $\Delta(M, \bar{M})$ is as in Eq. (3.2), except we use the "free-space" definition of IOU from [118], and account for the fact that orientation is only identifiable modulo 90° rotations. Because layout annotations do not necessarily have Manhattan structure, the ground truth layout is taken to be the cuboid hypotheses with largest free-space IOU.

Layout Hypotheses We predict floors and ceilings as the 0.001 and 0.999 quantiles of the 3D points along the gravity direction, and discretize orientation into 18 evenly spaced angles between 0 and 180°. We then propose layout candidates that capture at least 80% of all 3D points, and are bounded by the farthest and closest 3D points. For typical scenes, there are 5,000-20,000 layout hypotheses. See the supplemental material for more details.

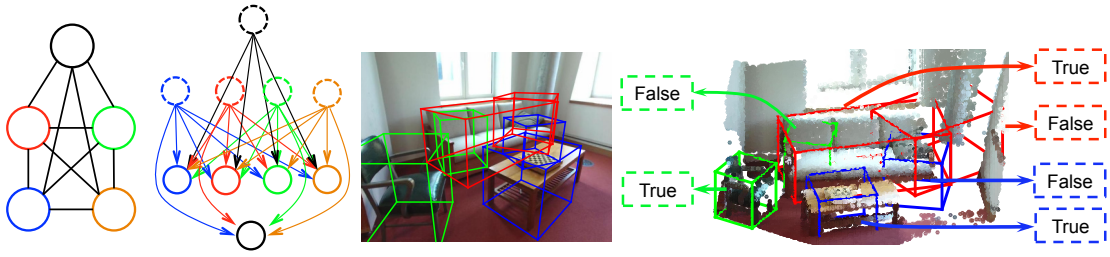


Figure 3.4: An illustration of how cascaded classification captures contextual relationships among objects. From left to right: (i) A traditional undirected MRF representation of contextual relationships. Colored nodes represent four object categories, and black nodes represent the room layout. (ii) A directed graphical representation of cascaded classification, where the first-stage detectors are hidden variables (dashed) that model contextual relationships among object and layout hypotheses (solid). Marginalizing the hidden nodes recovers the undirected MRF. (iii) First-stage detections independently computed for each category as in Sec. 3.2.2. (iv) Second-stage detections (Sec. 3.2.3) efficiently computed using our directed representation of context, and capturing contextual relationships between objects and the overall scene layout.

3.2.3 Cascaded Learning of Spatial Context

If the detectors learned in Sec. 3.2.2 are independently applied for each category, there may be many false positives, where a "piece" of a large object is detected as a smaller object (see Fig. 3.4). Song et al. [119] reduce such errors via a heuristic reduction in confidence scores for small detections on large image segments. To avoid such manual engineering, which must often be tuned to each category, we propose to directly learn the relationships among detections of different categories. As room geometry is also an important cue for object detection, we integrate Manhattan layout hypotheses for *total scene understanding* [118, 77].

Typically, structured prediction of spatial relationships is accomplished via undirected *Markov random fields* (MRFs) [87]. As shown in Fig. 3.4, this generally leads to a *fully connected* graph [93] because there are relationships among every pair of object categories. An extremely challenging MAP estimation (or energy minimization) problem must then be solved at every training iteration, as well as for each test image, so learning and prediction is costly.

We propose to instead adapt *cascaded classification* [52] to the modeling of contextual relationships in 3D scenes. In this approach, "first-stage" detections as in Sec. 3.2.2 become input features to "second-stage" classifiers that estimate confidence in the correctness of cuboid hypotheses. This

can be interpreted as a *directed* graphical model with hidden variables. Marginalizing the first-stage variables recovers a standard, fully-connected undirected graph. Crucially however, the cascaded representation is far more efficient: training *decomposes* into independent learning problems for each node (object category), and optimal test classification is possible via a rapid *sequence* of local decisions.

Contextual Features For an overlapping pair of detected bounding boxes B_i and B_j, we denote their volumes as $V(B_i)$ and $V(B_j)$, their volume of their overlap as $O(B_i, B_j)$, and the volume of their union as $U(B_i, B_j)$. We characterize their geometric relationship via three features: $S_1(i,j) = \frac{O(B_i,B_j)}{V(B_i)}$, $S_2(i,j) = \frac{O(B_i,B_j)}{V(B_j)}$, and the IOU $S_3(i,j) = \frac{O(B_i,B_j)}{U(B_i,B_j)}$. To model object-layout context [77], we compute the distance $D(B_i, M)$ and angle $A(B_i, M)$ of cuboid B_i to the closest wall in layout M.

The first-stage detectors provide a most-probable layout hypothesis, as well as a set of detections (following non-maximum suppression) for each category. For a bounding box B_i with confidence score z_i, there may be several overlapping bounding boxes of categories $c \in \{1, \ldots, C\}$. Letting i_c be the instance of category c with maximum confidence z_{i_c}, features ψ_i for bounding box B_i are created via a quadratic function of z_i, $S_{1:3}(i, i_c)$, $A(B_i, M)$, and a radial basis expansion of $D(B_i, M)$. Relationships between second-stage layout candidates and object cuboids are modeled similarly. See the supplemental material for details.

Contextual Learning Due to the directed graphical structure of the cascade, each second-stage detector may be learned independently. The objective is simple binary classification: is the candidate detection a true positive, or a false positive? During training, each detected bounding box for each class is marked as "true" if its intersection-over-union score to a ground truth instance is greater than 0.25, and is the largest among those detections. We train a standard binary SVM with a radial basis function (RBF) kernel

$$K(B_i, B_j) = \exp\left(-\gamma ||\psi_i - \psi_j||^2\right). \tag{3.3}$$

The bandwidth parameter γ is chosen using validation data. While we use a RBF kernel for all reported experiments, the performance of a linear SVM is only slightly worse, and cascaded classification still provides useful performance gains for that more scalable training objective.

										
Sliding-Shape [119]	42.95	19.66	20.60	28.21	60.89	-	-	-	-	-
Geom	8.29	15.06	26.20	24.53	1.15	-	-	-	-	-
Geom+COG	52.98	28.64	42.16	45.14	43.00	28.17	7.93	14.25	12.83	47.69
Geom+COG+Context-5	58.72	44.04	42.50	54.81	63.19	-	-	-	-	-
Geom+COG+Context-10	61.29	48.68	49.80	59.03	66.31	44.58	12.97	25.14	30.05	56.78
Geom+COG+Context-10+Layout	**63.67**	**51.29**	**51.02**	**62.17**	**70.07**	**45.19**	**15.47**	**27.36**	**31.80**	**58.26**

Table 3.1: Average precision scores for all object categories, from left to right: *bed, table, sofa, chair, toilet, desk, dresser, night-stand, bookshelf, bathtub*. Notice that using COG features without second-stage context already outperforms [119], training a second stage classifier with more contextual categories and room layout further boosts performance, and that [119] cannot model categories without CAD models.

To train the second-stage layout predictor (the bottom node in Fig. 3.4), we combine the object-layout features with the Manhattan voxel features from Sec. 3.2.1, and again use S-SVM training to optimize the free-space IOU.

Contextual Prediction During testing, given the set of cuboids found in the first-stage sliding-window search, we apply the second-stage cascaded classifier to each cuboid B_i to get a new contextual confidence score z_i'. The overall confidence score used for precision-recall evaluation is then $z_i + z_i'$, to account for both the original belief from the geometric and COG features and the correcting power of contextual cues. The second-stage layout prediction is directly provided by the second-stage S-SVM classifier.

3.2.4 Experiments

We test our cascaded model on the SUN RGB-D dataset [118] and compare with the state-of-the-art *sliding shape* [119] cuboid detector, and the baseline layout predictor from [118]. The older NYU Depth dataset [117] is a subset of SUN RGB-D, but SUN RGB-D has improved annotations and many new images. Since unlike prior work we do not use CAD models, we easily learn and evaluate RGB-D appearance models of 10 object categories, five more than [119]. Object cuboid and 3D layout hypotheses are generated and evaluated as described in previous sections.

We evaluate detection performance via the intersection-over-union with ground-truth cuboid

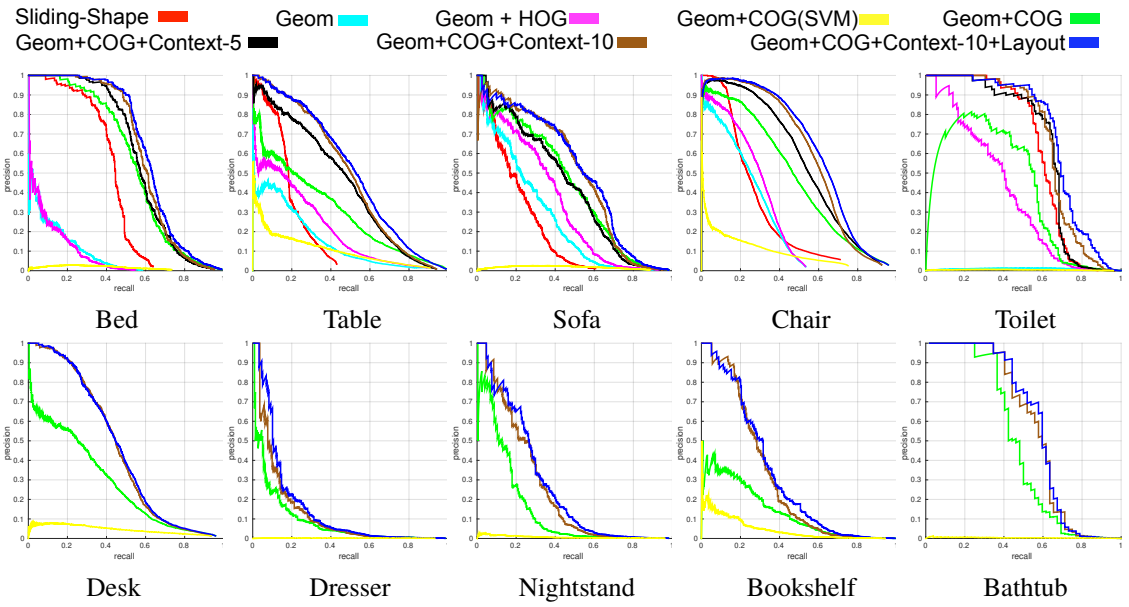


Figure 3.5: Precision-recall curves for 3D cuboid detection of the 5 object categories considered by [119] (top), and 5 additional categories (bottom). For the first 5 categories, we also test the importance of various features, and the gains from modeling context. See legend at top.

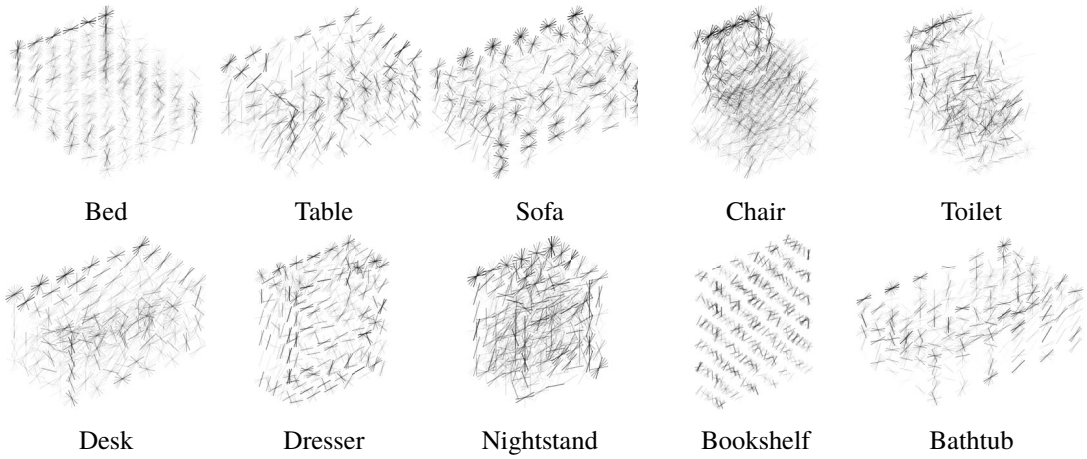


Figure 3.6: Visualization of the learned 3D COG features for all 10 categories. Reference orientation bins with larger weights are darker, and the 3D visualization is similar to each category's appearance. Cuboid sizes are set to the median of all training instances.

annotations, and consider the predicted box to be correct when the score is above 0.25. To evaluate the layout prediction performance, we calculate the free space intersection-over-union with human annotations. We provide several comparisons to demonstrate the effectiveness of our scene understanding system, and the importance of both appearance and context features.

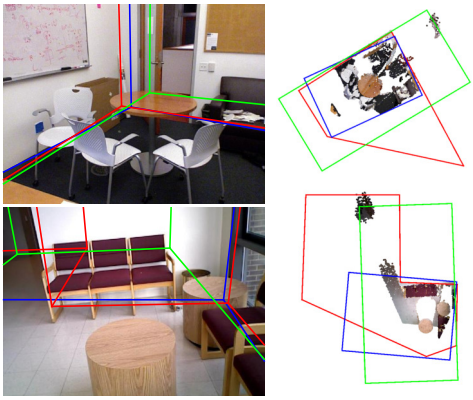


Figure 3.7: Comparison of our Manhattan voxel 3D layout predictions (blue) to the SUN RGB-D baseline ([118], green) and the ground truth annotations (red). Our learning-based approach is less sensitive to outliers and degrades gracefully in cases where the true scene structure violates the Manhattan world assumption.

The Importance of Appearance We trained our detector with geometric features only (Geom), and with the COG feature added (Geom+COG). There is a very clear improvement in detection accuracy for all object categories (see Table 3.1 and precision-recall curves in Fig. 3.5). Object detectors based solely on noisy point clouds are imperfect, and the RGB image contains complementary information.

HOG versus COG To demonstrate the effectiveness of the COG feature, we also use naïve 2D bins to extract HOG features for each 3D cuboid and train a detector (Geom+HOG). Since fixed 2D bins do not align with changes in 3D object pose, this feature is less informative, and detection performance is much worse than when using COG bins corrected for perspective projection.

We visualize the learned COG features for different categories in Fig. 3.6. We can see many descriptive appearance cues such as the oriented exterior boundaries of each object, and hollow regions for sofa, chair, toilet, and bathtub.

Cubical Voxels versus Manhattan Voxels We use the free-space IOU [118] to evaluate the performance of layout prediction algorithms. Using standard cubical voxels, our performance (**72.33**) is similar to the heuristic SUN RGB-D baseline (**73.4**, [118]). Combining Manhattan voxels with structured learning, performance increases to **78.96**, demonstrating the effectiveness of this improved discretization. Furthermore, if we also incorporate contextual cues from detected objects, the score improves to **80.23**. We provide some layout prediction examples in Fig. 3.7.

	P_g	R_g	R_r	IoU
Sliding-Shape+Plane-Fitting [118]	37.8	32.3	23.7	66.0
COG+Manhattan Voxel+Context	**47.3**	**36.8**	**35.8**	**72.0**

Table 3.2: Evaluation of total scene understanding [118]. We choose a threshold for object confidence scores that maximizes P_g, and compute all other metrics. Our highly accurate object and layout predictions also lead to improved overall scene interpretations.

The Importance of Context To show that the cascaded classifier helps to prune false positives, we evaluate detections using the confidence scores from the first-stage classifier, as well as the updated confidence scores from the second-stage classifier (Geom+COG+Context-5). As shown in Table 3.1 and Fig. 3.5, adding a contextual cascade clearly boosts performance. Furthermore, when more object categories are modeled (Geom+COG+Context-10), performance increases further. This result demonstrates that even if a small number of objects are of primary interest, building models of the broader scene can be very beneficial.

We show some representative detection results in Fig. 3.8. In the first image our chair detector is confused and fires on part of the sofa, but with the help of contextual cues of other detected bounding boxes, these false positives are pruned away. For a fixed threshold across all object categories, we have as many true detections as the sliding-shape baseline while producing fewer false positives.

Total Scene Understanding By capturing contextual relationships between pairs of objects, and between objects and the overall 3D room layout, our cascaded classifier enables us to perform the task of total scene understanding [118]. We generate a single global scene hypothesis by applying the same threshold (tuned on validation data) to all second-stage object proposals, and choose the highest-scoring layout prediction. We report the precision, recall, and IOU evaluation metrics defined by [118] in Table 3.2. In every case, we show clear improvements over baselines.

Computation Speed Our algorithm, implemented in MATLAB, spends most of its running time on feature computation. For a typical indoor image, our algorithm will spend 10 to 30 minutes to compute features for one object category and Manhattan Voxel discretization, and 2 seconds to predict 3D cuboids and layout hypotheses. This speed could be dramatically improved in various ways, such as exploiting integral images for feature computation [119] or using GPU hardware for parallelization.

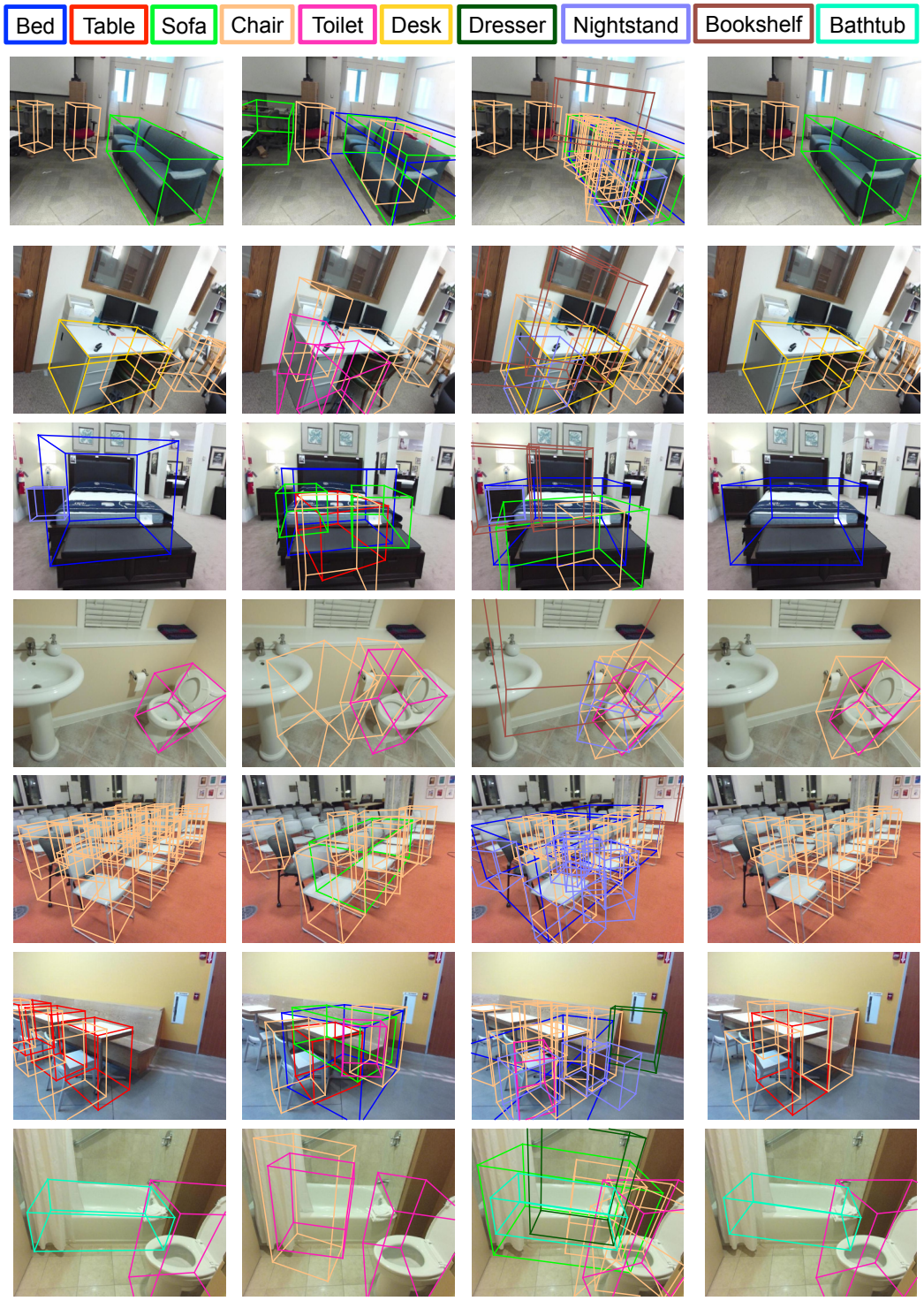


Figure 3.8: Detections with confidence scores larger than the same threshold for each algorithm. Notice that using contextual information helps prune away false positives and preserves true positives.

3.3 3D Object Detection with Latent Support Surfaces

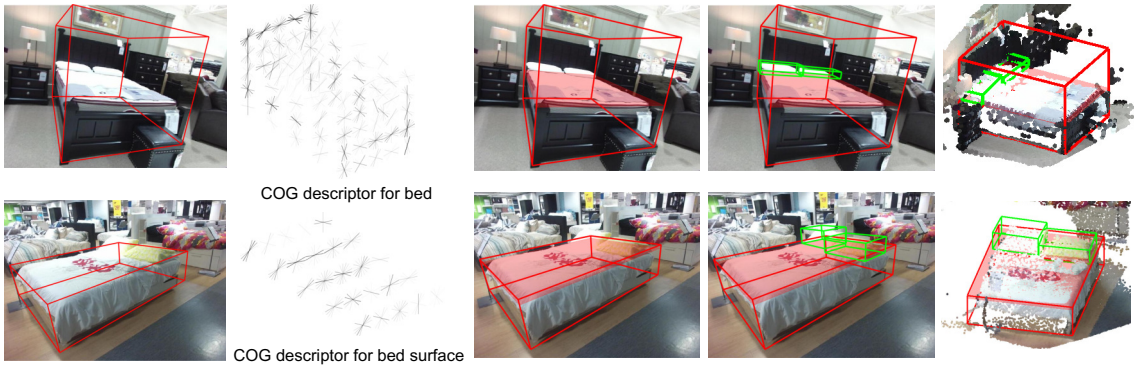


Figure 3.9: A visualization of 3D object detection system for beds and pillows using latent support surfaces. Given input RGB-D images, we use our learned COG descriptor [99] to localize 3D objects and infer latent support surfaces (shaded) for 3D proposals of beds (red). Then we search for pillows (green) that lie on top of the inferred support surfaces.

3.3.1 Effective Extensions to COG Descriptor

Feature extraction is one of the most important steps for object detection algorithms. 2D object detectors typically use either hand-crafted features based on image gradients [27, 33] or learned features from deep neural networks [43, 42, 97, 49, 78]. For 3D object detection systems with additional depth inputs, Gupta *et al*. [48] use horizontal disparity, height above ground, and the angle of pixel's local surface normal to encode images as a three channel (HHA) map for input to a convolutional neural network. While such convolutional processing of 2D images may be used to extract features from 2D bounding boxes, it does not directly provide a method for recovering 3D bounding boxes.

Song *et al*. [119] use 3D truncated signed distance function (TSDF) features to encode 3D cuboids, and their subsequent deep sliding shape [120] method aggregates TSDF with standard 2D features from a deep convolutional neural network. However, those features do not explicitly capture 3D orientation. We instead build our 3D detection algorithm on the *cloud of oriented gradients* (COG) descriptor [99]. We introduce simple extensions that improve its performance.

View-to-Camera Feature For single view RGB-D inputs, an object like nightstand may only expose one planer surface to the camera. At test time, features of a 3D cuboid proposal whose orientation is facing backwards resembles those of a correct detection (Fig. 3.10). This is because voxel features are computed by first rotating the cuboid to a canonical coordinate frame. However, due to the self-occlusions that occur in real objects, the features modeled by the COG descriptor would in fact not be visible when objects are facing away from the camera. Therefore, we add features to represent objects' view to camera, and learn to explicitly distinguish implausible object orientations.

Figure 3.10: A false positive 3D detection for nightstand without using view-to-camera feature (left). The COG feature is similar to that of a correct detection (right) but the orientation is flipped.

Specifically, we compute the cosine x of the angle between the cuboid orientation and its viewing angle from camera in horizontal direction. Then we define a set of radial basis functions of the form

$$f_j(x) = \exp\Big(-\frac{(x-\mu_j)^2}{2\sigma^2}\Big),$$

and space the basis function centers μ_j evenly between $[-1, 1]$ with step size 0.2. The bandwidth $\sigma = 0.5$ was chosen using validation data. Radial basis expansions are a standard approach to non-linear regression, and can also be seen as a layer of a neural network. We expand the camera angle using this basis representation and refer to the resulting 11-dimensional vector as the *view-to-camera feature*.

Scene Layout Feature The interaction between objects and the scene layout (floor, walls, ceiling) provides important cues for object detection. For example, Song *et al*. [120] propose 3D objects along the predicted walls of the scene. Ren *et al*. [99] introduce a Manhattan voxel discretization to better predict scene layouts and model object-layout interactions. We model object-layout interactions by first computing the distance and angle to the nearest predicted wall using Manhattan voxels [99], then expand the distance-to-wall value using radial basis functions spaced between $[0, 5]$ with step size 0.5. We also expand the absolute cosine value of the angle-to-wall using radial basis functions spaced between $[0, 1]$ with step size 0.2 and $\sigma = 0.5$. Combining these layout features with the view-to-camera feature, we are able to improve detection performance for most object categories (see Table 3.3).

3.3.2 Modeling Latent Support Surfaces

Geometric descriptors and COG descriptors [99] are able to capture local shapes and appearances. However, 3D objects in indoor scenes have widely varying visual styles. Moreover, 3D cuboid annotations are labeled by different people from Mechanical Turk in SUN RGB-D dataset [118], thus objects in the same category may have inconsistent 3D annotations. As a result, features are inevitably noisy and inconsistent across different object instances (see Fig. 3.11).

To explicitly model different visual styles for each objects, a classical approach is to use part-based models [33, 35] where objects are explained by spatially arranged parts. However, indoor objects have very diverse visual styles, and it is very challenging to design a consistently varying set of latent parts. However, for many object categories, the height of the support surface is the primary cause of style variations (Fig. 3.11). Therefore, we explicitly model the support surface as a latent part for each object.

By modeling support surfaces we can also constrain the search space for small object detectors. Such detectors are otherwise intractable to learn and perform poorly due to the large set of possible 3D poses [99].

Figure 3.11: Different surface heights for "desk" in SUN RGB-D dataset [118] lead to inconsistent 3D COG representations [99].

Latent Structural SVM Learning

Some previous work was specifically designed to predict the area of support surface regions [45], but the predicted support surfaces are not semantically meaningful. Inspired by deformable part-based models for 2D object detection [33], we propose to treat the relative height of the support surface of each object as a latent variable and use latent structural SVMs [148] to learn the detector.

We follow the notation of Ren *et al*. [99] with an updated learning objective. For each category c, our goal is to learn a prediction function $I \rightarrow (B, h)$ that maps an RGB-D image I to a 3D bounding box $B = (L, \theta, S, y)$ along with its relative surface height h. L is the center of the cuboid in 3D, θ is the cuboid orientation, S is the physical size of the cuboid along the three axes determined by its orientation, and y is an indicator variable representing the existence of such prediction. The latent variable h is defined as the relative surface height to the bottom of the cuboid. We discretize cuboid height to 7 slices, and thus h localizes the support surface to one of those slices (see Fig. 3.12).

Given n training examples of category c, we want to solve the following optimization problem:

$$\min_{w_c, \xi \geq 0} \quad \frac{1}{2} w_c^T w_c + \frac{C}{n} \sum_{i=1}^{n} \xi_i \quad \text{subject to}$$

$$\max_{h_i \in \mathcal{H}} w_c^T \phi(I_i, B_i, h_i) - \max_{\bar{h}_i \in \mathcal{H}} w_c^T \phi(I_i, \bar{B}_i, \bar{h}_i)$$

$$\geq \Delta(B_i, \bar{B}_i, \bar{h}_i) - \xi_i, \quad \text{for all } \bar{B}_i \in \mathcal{B}_i, i = 1, \ldots, n.$$

Here B_i is the ground-truth bounding box, $\mathcal{B}_i$ is the set of possible bounding boxes, and $\mathcal{H}$ is the set of possible surface heights. $\phi(I, B, h)$ are the features associated to cuboid B whose relative surface height is indicated by h. We first discretize B into $5 \times 5 \times 5$ voxels and compute geometric features, COG [99], view-to-camera feature, and scene layout feature, as denoted by $\phi_{\text{cuboid}}(I, B)$. Then we discretize B with finer resolutions at the vertical dimension into $5 \times 5 \times 7$ voxels and take the h-th slice from the bottom to represent cuboid feature, as denoted by $\phi_{\text{surface}}(I, B, h)$. Finally we add an indicator vector for support surface height, so that

$$\phi(I, B, h) = [\phi_{\text{cuboid}}(I, B), \phi_{\text{surface}}(I, B, h), 0, ..., 1, ..., 0].$$

If the indicator variable y in B is 0, meaning there's no detection, we set the feature vector to be all zeros. A visualization of support surface feature is shown in Fig. 3.12.

Following Ren *et al*. [99] we define a loss function for cuboid proposals $\bar{B}$: If a scene contain

ground truth cuboid B and indicator variable $\bar{y}$ is 1, we compute

$$\Delta(B, \bar{B}) = 1 - \text{IOU}(B, \bar{B}) \cdot \left(\frac{1 + \cos(\bar{\theta} - \theta)}{2} \right).$$

where $\text{IOU}(B, \bar{B})$ is 3D intersection over union. The scale of this loss function ranges in $[0, 1]$. If a scene doesn't contain any ground truth cuboid and the indicator variable $\bar{y}$ is 0 for the cuboid proposal, the loss is set to be 0. We penalize all other cases with a loss of 1.

To train the model with latent support surfaces, we follow Ren *et al*. [99] by pre-training cuboid descriptors (geometric features, COG, view-to-camera, and scene layout feature) without modeling support surface. We then extract the center slice of pre-trained cuboid descriptors and concatenate it to the pre-trained models. Finally, we initialize the support surface height indicator vector randomly in $[0, 1]$. We use the CCCP algorithm [149] to solve the resulting latent structural SVM learning problem [148].

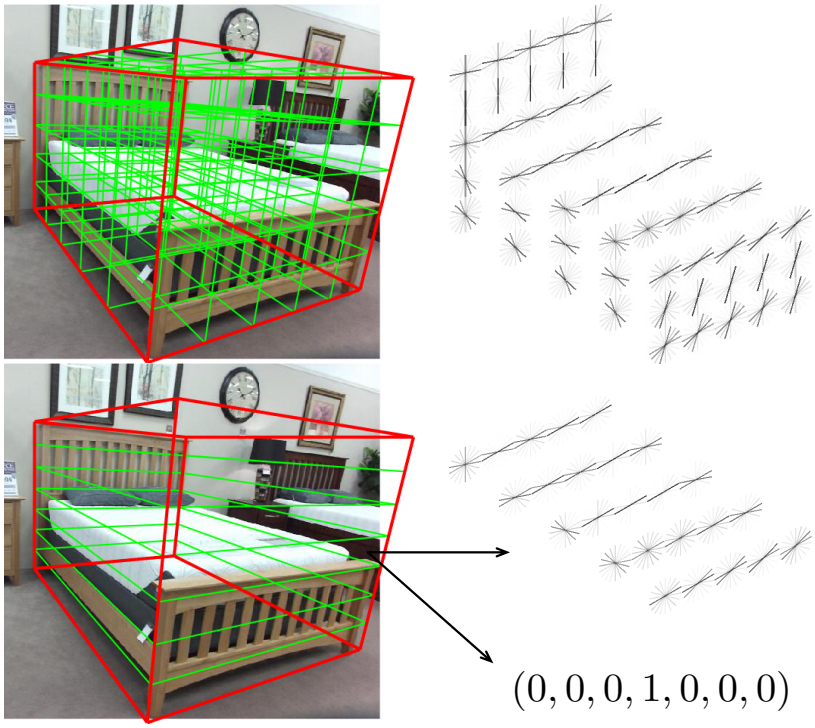


Figure 3.12: Features of 3D cuboid with support surface. The surface feature is computed at a single slice of the cuboid followed by an indicator vector to represent the relative height.

Small Object Detection on Support Surfaces

In indoor scenes, besides large furniture like beds and chairs, many other objects with comparatively small physical size are very hard to detect [120, 99]. Some algorithms are specifically designed to detect small objects in 2D images using multi-scale methods [17, 55], but they cannot be directly applied to 3D object detection.

The biggest issue for detecting small objects is that the search space can be enormous, and thus training and testing with a sliding-windows based approach are usually intractable. But note that small objects, such as pillows and monitors and lamps, are usually placed on top of other objects with support surfaces. If we only search for small objects on predicted support surfaces, the search space will be greatly reduced. As a result, the inference speed will be improved and object proposals contain less false positives. This is another benefit of modeling support surfaces.

In our implementation, we first detect large objects of indoor scenes that are on the ground [99], then we search for smaller objects only on top of the support surfaces of those large objects with positive confidence scores. We reduce the voxel descritization size to be $3 \times 3 \times 3$ for lamps and pillows because small cuboids contain less pixels, and $3 \times 1 \times 3$ for monitors and TVs because they have thin shapes.

Spatial Contextual Learning for All Objects

Our object detector is trained discriminatively for each object category. At test time, 3D objects with locally similar shapes can confuse 3D detectors trained for each object category independently. Instead of designing simple heuristics to handle false positives, we follow Ren *et al*. [99] by using an effective cascaded detection framework [52] to model contextual relationships among cuboid proposals.

For each 3D cuboid proposal, we encode its contextual relationship with the highest confidence cuboid proposals in all object categories using 3D overlapping features and confidence differences. Using those contextual features we learn a linear SVM to determine whether those 3D object proposals are correct or not, and add this updated confidence score to first-stage detection scores. We refer readers to the supplementary material of Ren *et al*. [99] for a detailed explaination. For

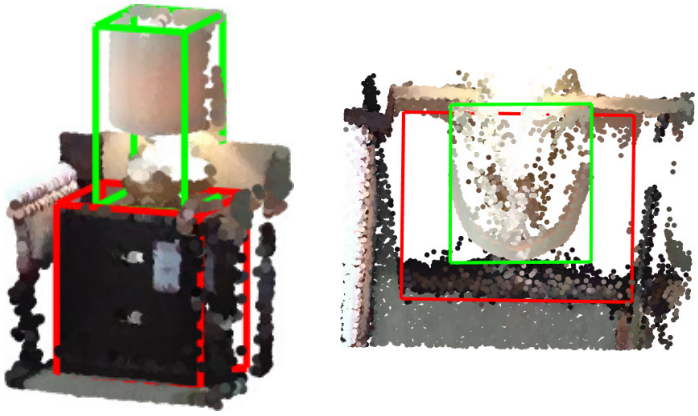


Figure 3.13: To model contextual relationships between small objects and the large objects supporting them, we compute the 2D overlap between 3D bounding boxes from the top-down view.

small objects that are placed on the support surfaces of large objects, 3D overlap features are noisy. We replace 3D overlap with 2D overlap scores from the top-down view of the scene (Fig. 3.13). With updated confidence scores that account for both original beliefs and contextual cues, object proposals contain fewer false positives and object detectors have improved performance.

	Bathtub	Bed	Bookshelf	Chair	Desk	Dresser	Nightstand	Sofa	Table	Toilet	Box	Door	Counter	Garbage-bin	Sink	Pillow	Monitor	TV	Lamp	mAP (10)	mAP (19)
COG	49.8	53.0	8.5	39.0	14.9	5.5	12.8	52.8	26.0	34.5	11.6	0.9	2.4	20.1	30.3	0.0	0.0	0.0	0.0	29.68	19.1
+view	60.2	59.3	18.4	40.4	18.0	9.5	16.8	53.2	26.8	41.6	11.0	3.2	4.6	21.8	30.6	0.0	0.0	0.0	0.0	34.4	21.9
+surface	66.6	68.0	21.5	42.0	26.0	8.5	17.1	52.8	39.0	45.8	11.3	2.6	4.0	19.7	60.9	9.9	1.6	0.4	9.6	38.7	26.7
+cascade	**76.2**	73.2	**32.9**	60.5	34.5	13.5	30.4	**60.4**	**55.4**	73.7	**19.5**	**5.4**	**10.7**	**34.6**	**75.3**	12.5	**1.6**	**2.1**	16.9	**51.0**	**36.3**
SS [119]	-	43.0	-	28.2	-	-	-	20.6	19.7	60.9	-	-	-	-	-	-	-	-	-	-	-
DSS [120]	44.2	**78.8**	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	1.5	0.0	4.1	20.4	32.3	**13.3**	0.2	0.5	**18.4**	42.1	26.9
Ren [99]	58.3	63.7	31.8	**62.2**	**45.2**	15.5	27.4	51.0	51.3	70.1	-	-	-	-	-	-	-	-	-	47.6	-
Lahoud [69]	43.5	64.5	31.4	48.3	27.9	**25.92**	**41.9**	40.39	37.0	**80.4**	-	-	-	-	-	-	-	-	-	45.1	-

Table 3.3: Experiment results on SUN RGB-D dataset [118]. Our baseline method uses COG descriptor. Adding extra features to model view-to-camera and scene layout (+view) improves performance, and modeling support surfaces (+surface) not only help detect large objects but also reduce many false positives for small objects (last 4 categories). The final stage cascaded detection framework [99] (+cascade) models object context and help boost the performance to the state-of-the-art over existing methods for the first 10 and all 19 object categories.

	Bathtub	Bed	Bookshelf	Chair	Desk	Dresser	Nightstand	Sofa	Table	Toilet	Box	Door	Counter	Garbage-bin	Sink	Pillow	Monitor	TV	Lamp	mAP (10)	mAP (19)
Whole System	**76.2**	**73.2**	**32.9**	**60.5**	**34.5**	**13.5**	**30.4**	**60.4**	**55.4**	**73.7**	**19.5**	**5.4**	**10.7**	34.6	**75.3**	**12.5**	**1.3**	**2.1**	**16.9**	**51.0**	**36.3**
-view-surface	53.3	63.0	18.7	61.6	29.0	7.5	20.2	58.8	49.1	62.8	17.3	1.1	6.6	**39.1**	60.3	0.0	0.0	0.0	0.0	42.4	28.9

Table 3.4: We compare our holistic scene understanding system with cascaded detection on SUN-RGBD dataset [118]. Although cascaded detection is powerful, there is still a drop in performance without modeling view-to-camera feature, scene layout and support surfaces.

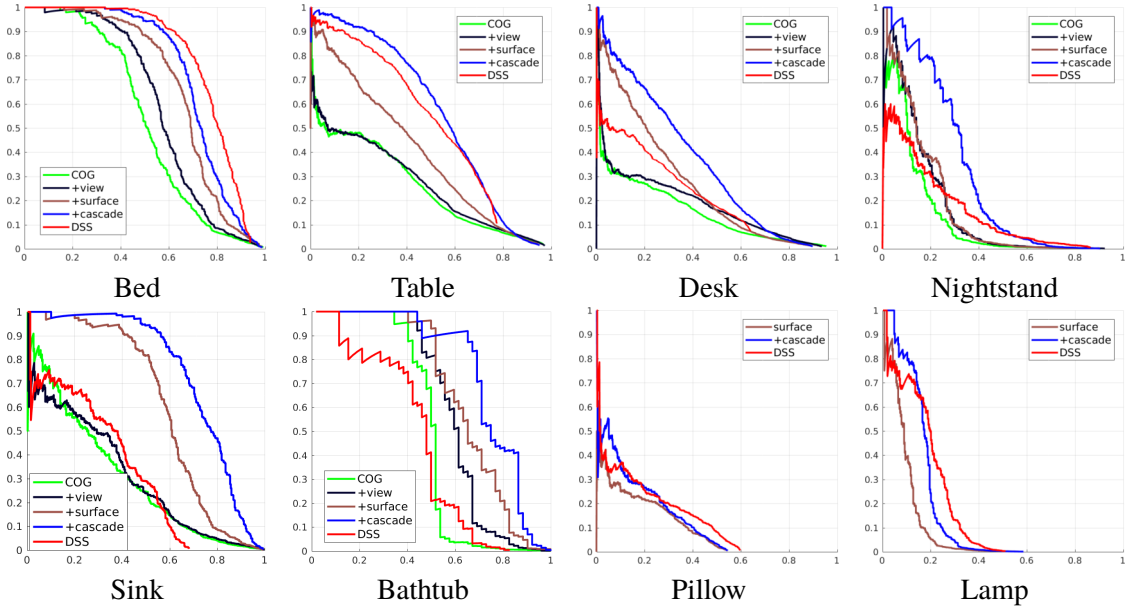


Figure 3.14: Precision-Recall curves for several object categories including small objects (pillow and lamp) on SUN RGB-D dataset [118].

3.3.3 Experiments

We train our 3D object detection algorithm solely on the SUN RGB-D dataset [118] with 5285 training images, and report performance on 5050 test images for all 19 object categories (Table 3.3). The NYU Depth dataset [117] has 3D cuboid labels for 1449 images, but annotations are noisy and inconsistent. Some previous work has only evaluated detection performance on this small dataset [47], or defined their own annotations for 3D cuboids [29]. Because the SUN RGB-D dataset contains all images from NYU Depth dataset with more accurate annotations, we do not evaluate on the NYU Depth dataset in this chapter.

Baseline Algorithm using COG We implement a baseline detector using only COG features [99] and local geometric features of the point cloud. This method is denoted by "COG" in the first row of Table 3.3. Note that this detectors' performances is slightly different from the first stage detection scores in Ren *et al*. [99] because we are using a coarser $5 \times 5 \times 5$ discretization (versus $6 \times 6 \times 6$) for each cuboid. With reduced feature size, our algorithm is more computationally efficient but has similar accuracy.

Effectiveness of View-based Features By adding extra view-to-camera features and scene layout features, denoted by "+view" in the second row of Table 3.3, we witness notable improvements on detecting small objects with a layered shape such as dressers and nightstands. Those objects usually expose only one side to the camera, and the view-to-camera feature is helpful in distinguishing correct predictions. With scene layout features, we also witness improvements for objects whose orientations strongly correlate with directions of walls, such as beds and bookshelves.

Modeling Latent Support Surfaces For objects such as beds, tables, and desks, modeling support surface as a latent variable help capture the intra-class style variations within each cuboid and we witness great performance gains in the third row of Table 3.3. We visualize examples of inferred support surfaces in Figure 3.15. For objects that do not have explicit "support surfaces", such as bathtub, bookshelf, and sink, our model can be viewed as a single part-based model and is also effective for 3D object detection. Note that the goal of this work is to model latent support surface in order to help 3D detection, not to predict accurate support surface area of the scene. We do not use any annotations of support surfaces when training, and also do not evaluate our performance on surface prediction benchmarks [45].

Small Object Detection Detecting small objects is a challenging task and is still an open problem. Without modeling support surfaces, our baseline method fails to detect small objects because the search space is large and 3D object proposals contain many false positives. Using simple heuristics to check support relationships in the SUN-RGBD annotations, we find more than 95% of lamps/pillows/monitors/TVs are placed on the surface of night-stands/tables/beds/desks/dressers. Searching on the predicted surface region enables our algorithm to discover small objects with higher precision. See row 3 in Table 3.3.

Comparison to Other Methods By modeling latent support surface, our algorithm already outperforms the state-of-the-art method of Ren *et al*. [99] for 10 large object categories, and can also detect some smaller objects. In this chapter our main goal is to demonstrate the effectiveness of modeling latent support surface, and we think the current system already shows great potential.

Comparing with other algorithms that use CNN features [120, 69] pretrained on external datasets, the performance of our algorithm is comparable even without the cascaded prediction step. Conventional CNNs for 3D detection [120, 69] are trained to produce weighted confidence scores for each of multiple object categories, while our first-stage detector algorithm is instead tuned to discriminatively localize individual categories in 3D. Our subsequent cascaded prediction [52] of contextual relationships between object detections has structural similarities to a multi-stage neural network, but it is trained using (convex) structural SVM loss functions and designed to have a more interpretable, graphical structure. Interestingly, our cascaded approach is comparable to or more accurate than standard 3D CNNs [120, 69] in the detection of both 10 and 19 object categories.

Computational Speed We implemented our algorithm using MATLAB in a 2.5GHz single core CPU. The computational speed of our algorithm is 10-30min per image, which is slightly better than the reported speed in Ren *et al*. [99]. The most time-consuming part is the feature computation step, which could be improved by using parallel computing with multi-core CPUs or GPUs. With pre-computed cuboid features for each RGB-D image, the inference time is 2sec for each object category. With pre-computed contextual features among all objects, the cascaded prediction framework takes less than 0.5sec on average to run.

Failure Cases An typical failure case for our algorithm is shown in the last row of Fig. 3.15, where missing depth values cause objects to be missed. While it is true that some small objects will be missed when we fail to detect their supporting surface, given the extreme difficulty of detecting small objects in highly cluttered indoor scenes, there are still substantial net benefits to exploiting support relationships for 3D detection. Some previous work was specifically designed to solve this issue [29, 69] by using CNN features in RGB images, and we believe incorporating a similar approach in our cascaded prediction framework might also help resolve this failure case.

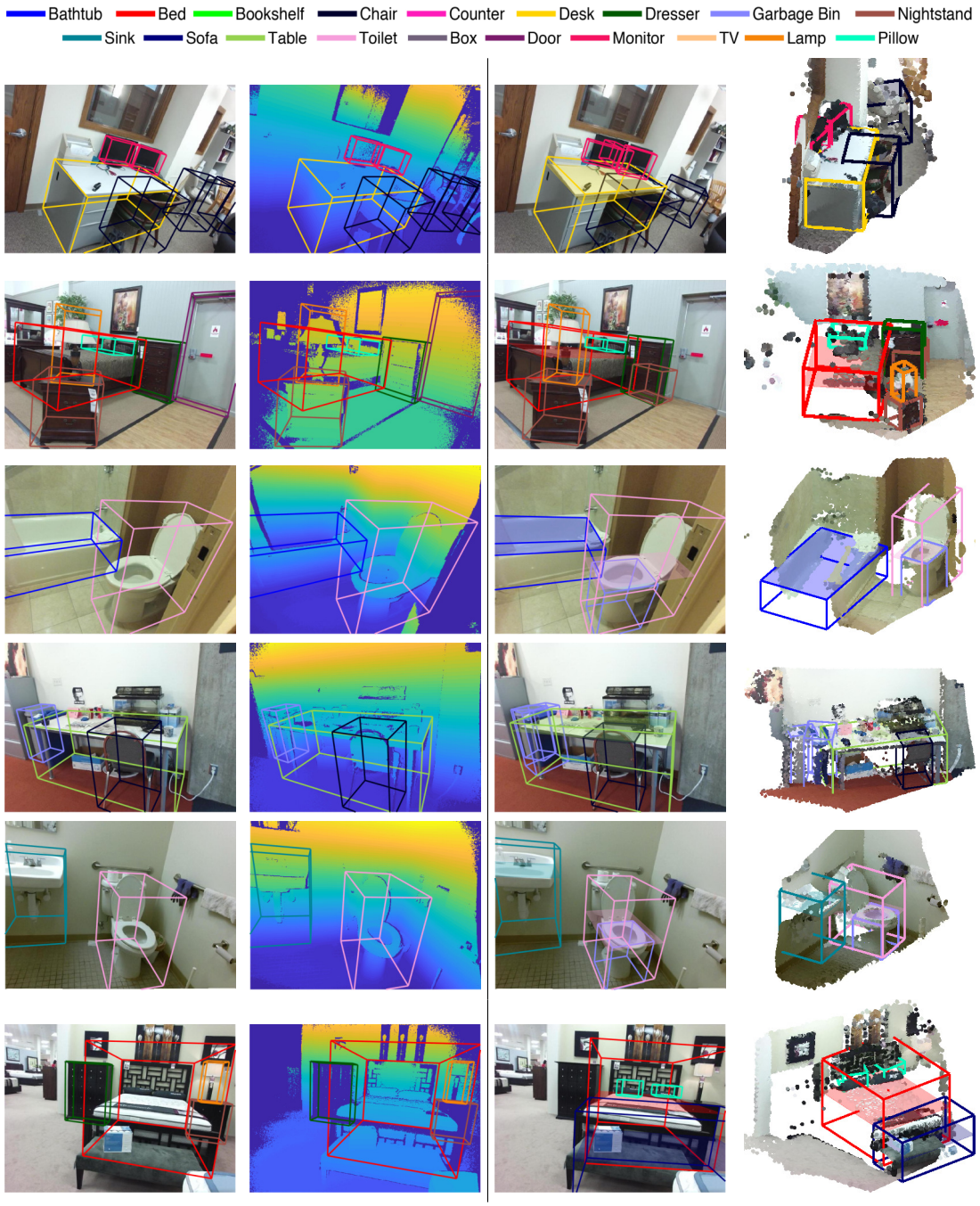


Figure 3.15: Visualizing our final stage 3D detections for objects with high confidence scores. Support surfaces are depicted with faded colors inside each large object. We show one failure case at the bottom: our algorithm failed to detect a dresser and a nightstand due to missing depth inputs (dark blue). As a result, the lamps supported by those objects are missed as well.

3.4 Conclusions

We propose an algorithm for 3D cuboid detection and Manhattan room layout prediction from RGB-D images. Using our novel COG descriptor of 3D appearance, we trained accurate 3D cuboid detectors for ten object categories, as well as a cascaded classifier that learns contextual cues to prune false positives. Our scene representations are learned directly from RGB-D data without external CAD models, and may be generalized to many other categories.

We then design a 3D object detection system using latent support surfaces. Modeling the height of the support surface as a latent variable leads to improved detection performance for large objects, and constrains the search space for small object detectors. Our detector achieves state-of-the-art performance on the SUN RGB-D dataset, demonstrating the effectiveness of modeling support surfaces in 3D object detection.

3.5 Supplementary Material

3.5.1 Proposing Layout Candidates

We predict floors and ceilings as the 0.001 and 0.999 quantiles of the 3D points along the gravity direction. After that, we only need to propose wall candidates that follows Manhattan structure.

We discretize orientation into 18 evenly spaced angles between 0 and 180°. For each orientation, the frontal wall is bounded by the 0.99 quantiles of the farthest points and the back wall is bounded by camera location. Because the layout candidates should follow a Manhattan structure, the left/right wall should be orthogonal to the frontal wall and should be bounded by the 3D points. We further discretize along the width and depth direction by 0.1m, and propose candidates that capture at least 80% of all 3D points. For typical scenes, there are 5,000-20,000 layout hypotheses.

3.5.2 Contextual Features

Here, we give a detailed explanation of the contextual features we use to model object-object and object-layout relationships in the second stage cascaded classifier.

For completeness, we define the notations again. For an overlapping pair of detected bounding boxes B_i and B_j, we denote their volumes as $V(B_i)$ and $V(B_j)$, their volume of their overlap as $O(B_i, B_j)$, and the volume of their union as $U(B_i, B_j)$. We characterize their geometric relationship via three features: $S_1(i,j) = \frac{O(B_i,B_j)}{V(B_i)}$, $S_2(i,j) = \frac{O(B_i,B_j)}{V(B_j)}$, and the IOU $S_3(i,j) = \frac{O(B_i,B_j)}{U(B_i,B_j)}$. To model object-layout context [77], we compute the distance $D(B_i, M)$ and angle $A(B_i, M)$ of cuboid B_i to the closest wall in layout M.

The first-stage detectors provide a most-probable layout hypothesis, as well as a set of detections (following non-maximum suppression) for each category. For each bounding box B_i with confidence score z_i, there may be several bounding boxes of various categories $c \in \{1, 2, ..., C\}$ that overlap with it. We let i_c be the instance of category c with the maximum confidence score z_{i_c}. The features ψ_i for bounding box B_i are then as follows:

1. Constant bias feature, and confidence score z_i from the first-stage detector.

2. For $m \in \{1, 2, 3\}$ and $c \in \{1, 2, ..., C\}$, we calculate $S_m(i, i_c)$, $S_m(i, i_c) \cdot z_{i_c}$, $S_m(i, i_c) \cdot z_i$ and concatenate those numbers.

3. For $c \in \{1, 2, ..., C\}$, we calculate the difference in confidence score from each first-stage detector, $z_i - z_{i_c}$, and concatenate those numbers.

4. For $D(B_i, M)$, we consider radial basis functions of the form

$$f_j(x) = \exp(-\frac{(x - \mu_j)^2}{2\sigma^2}) \tag{3.4}$$

   For a typical indoor scene, the largest object-to-wall distance is usually less than 5m, therefore we space the basis function centers μ_j evenly between 0 and 5 with step size 0.5, and choose $\sigma = 0.5$. We expand $D(B_i, M)$ using this radial basis expansion.

5. The absolute value of the cosine of $D(B_i, M)$: $|\cos(D(B_i, M))|$

To model the second-stage layout candidates, we select the bounding box i_c with the highest confidence score z_{i_c} from the first-stage classifier in each category $c \in \{1, 2, ..., C\}$, and use the following features for layout M_i with confidence score z'_i:

1. All the features used in the first-stage to model M_i using *Manhattan Voxels*.

2. For $c \in \{1, 2, ..., C\}$, we calculate the radial basis expansion for $D(B_{i_c}, M_i)$, and its product with z'_i and z_{i_c}.

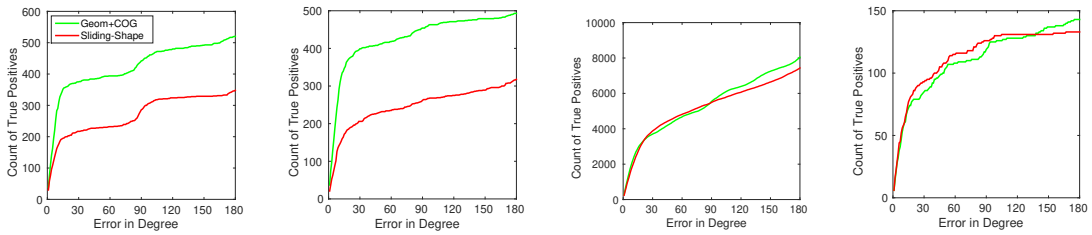
3. For $c \in \{1, 2, ..., C\}$, we calculate the absolute value of the cosine of $D(B_{i_c}, M_i)$: $|\cos(D(B_{i_c}, M_i))|$, $|\cos(D(B_{i_c}, M_i))| \cdot z'_i$ and $|\cos(D(B_{i_c}, M_i))| \cdot z_{i_c}$.

4. For $c \in \{1, 2, ..., C\}$, we calculate the difference in confidence score from each first-stage detector, $z'_i - z_{i_c}$, and concatenate those numbers.

3.5.3 Additional Experimental Results

Orientation Besides evaluating detection based on intersection-over-union score, we can also evaluate the accuracy of the predicted orientations. We plot the cumulative counts of true positive detections whose orientation error in degrees is less than certain thresholds in Fig. 3.16. Using

geometric feature and COG, our detector is able to detect more true positives than sliding-shape [119], while maintaining comparable accuracy in orientation estimation. Those curves are not related to the precision-recall curves as we are only evaluating on true positives.



Bed Sofa Chair Toilet

Figure 3.16: Quantification of orientation estimation accuracy for the four object categories considered by [118].

Additional Layout Prediction results Besides the two examples for layout prediction shown in this chapter, in figure 3.17 we show some additional results to demonstrate the effectiveness of *Manhattan Voxels*.

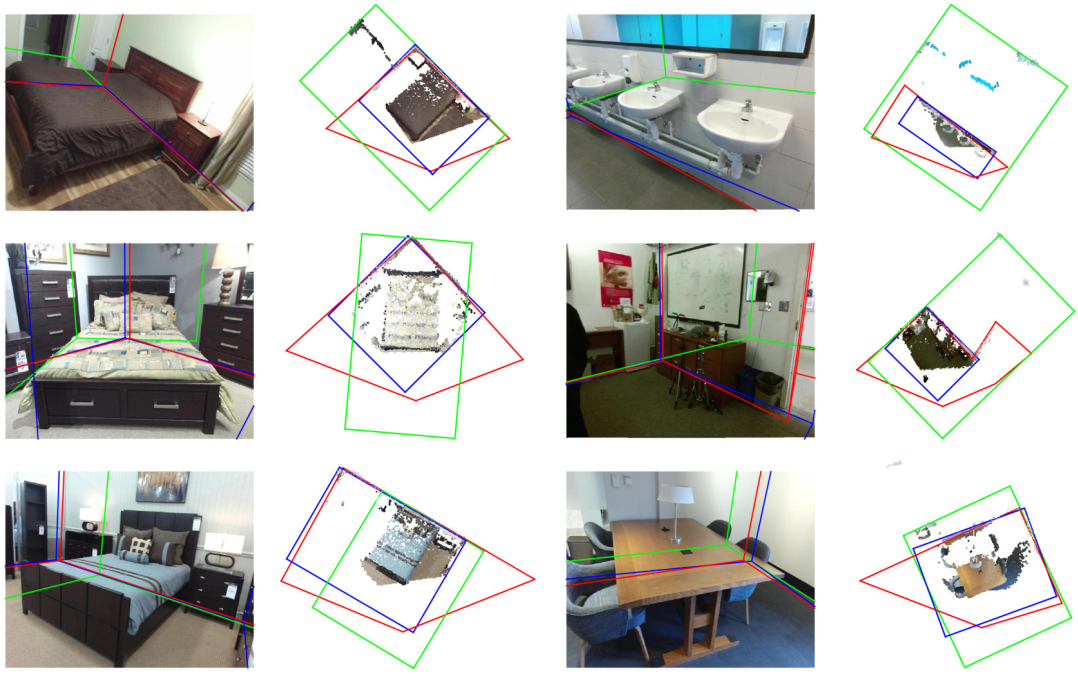


Figure 3.17: Comparison of our Manhattan voxel 3D layout predictions (blue) to the SUN RGB-D baseline ([118], green) and the ground truth annotations (red). Our learning-based approach is less sensitive to outliers and degrades gracefully in cases where the true scene structure violates the Manhattan world assumption.

Chapter 4

Cascaded Scene Flow Estimation using Semantic Segmentation

Given two consecutive frames from a pair of stereo cameras, 3D scene flow methods simultaneously estimate the 3D geometry and motion of the observed scene. Many existing approaches use superpixels for regularization, but may predict inconsistent shapes and motions inside rigidly moving objects. We instead assume that scenes consist of foreground objects rigidly moving in front of a static background, and use semantic cues to produce pixel-accurate scene flow estimates. Our cascaded classification framework accurately models 3D scenes by iteratively refining semantic segmentation masks, stereo correspondences, 3D rigid motion estimates, and optical flow fields. We evaluate our method on the challenging KITTI autonomous driving benchmark, and show that accounting for the motion of segmented vehicles leads to state-of-the-art performance.

4.1 Introduction

The *scene flow* [130] is the dense 3D geometry and motion of a dynamic scene. Given images captured by calibrated cameras at two (or more) frames, a 3D motion field can be recovered by projecting 2D motion (optical flow) estimates onto a depth map inferred via binocular stereo matching. Scene flow algorithms have many applications, ranging from driver assistance [86] to 3D

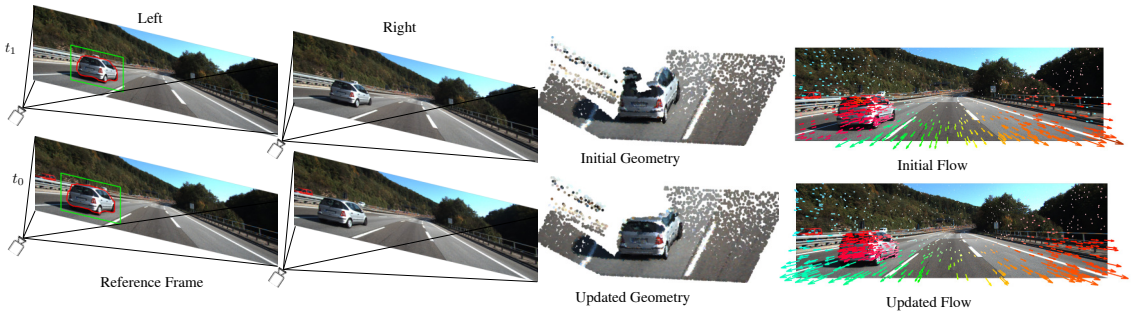


Figure 4.1: An illustration of our method for scene flow estimation. Given two frames from a pair of stereo cameras, and initial geometry and optical flow estimates provided by a non-semantic scene flow algorithm [133], we use semantic segmentation cues [26] to identify foreground vehicles. In this example, our updated geometry estimate reduces motion errors in the windshield of the car and the adjacent road.

motion capture [38].

The problems of optical flow estimation [124, 7] and binocular stereo reconstruction [108] have been widely studied in isolation. Recent scene flow methods [80, 146, 133] leverage 3D geometric cues to improve stereo and flow estimates, as evaluated on road scenes from the challenging KITTI scene flow benchmark [84]. State-of-the-art scene flow algorithms [134, 84] assume superpixels are approximately planar and undergo rigid 3D motion. Conditional random fields then provide temporal and spatial regularization for 3D motion estimates. Those methods generally perform well on background regions of the scene, but are significantly less accurate for moving foreground objects.

Estimating the geometry of rapidly moving foreground objects is difficult, especially near motion boundaries. Vehicles are particularly challenging because painted surfaces have little texture, windshields are transparent, and reflections violate the brightness constancy assumptions underlying stereo and flow likelihoods. However, accurate estimation of vehicle geometry and motion is critical for autonomous driving applications. To improve accuracy, it is natural to design models that separately model the motion of objects and background regions [89, 84].

Several recent methods for the estimation of optical flow [5, 57, 114, 89] have used semantic cues to improve accuracy. While motion segmentation using purely bottom-up cues is challenging, recent advances in semantic segmentation [154, 26] make it possible to accurately segment traffic

scenes given a single RGB image. Given segmented object boundaries, object-specific 3D motion models may then be used to increase the accuracy of optical flow methods.

In this paper, we use instance-level semantic segmentations [26] and piecewise-rigid scene flow estimates [133] as inputs, and integrate them via a cascade of *conditional random fields* (CRFs) [67]. We define pixel-level CRFs relating dense segmentation masks, stereo depth maps, optical flow fields, and rigid 3D motion estimates for foreground objects. Due to the high dimensionality of these variables, we refine them iteratively using a cascaded classification model [52], where each stage of the cascade is tuned via structural SVM learning algorithms [61]. We evaluate using previous scene flow annotations [84] of the challenging KITTI autonomous driving benchmark [40], and improve on the state-of-the-art in two-frame scene flow estimation. Our work demonstrates the importance of semantic cues in the recovery of the geometry and motion of 3D scenes.

4.2 Modeling Semantic Scene Flow

Given two consecutive frames I, J and their corresponding stereo pairs I', J', our goal is to estimate the segmentation mask, stereo disparity, and optical flow for each pixel in the reference frame (Fig. 4.1). Let $p_i = (d_i^{(1)}, s_i^{(1)}, m_i, f_i)$ denote the variables associated with pixel i in the reference frame, where $d_i^{(1)} \in \mathbb{R}^+$ is its disparity, $s_i^{(1)} \in \{0, 1, \ldots\}$ is a semantic label (0 is background, positive integers are foreground object instances), $m_i \in SE(3)$ is its 3D rigid motion (translation and rotation), and $f_i = [u_i, v_i]$ is its optical flow. We denote the disparity and semantic segmentation for each pixel in the second frame by $q_i = (d_i^{(2)}, s_i^{(2)})$. We only use two frames to estimate scene flow, and thus need not explicitly model motion in the second frame.

Existing scene flow algorithms make predictions at the superpixel level without explicitly modeling the semantic content of the scene [84, 134]. Predictions inside each semantic object may thus be noisy or inconsistent. In this work, we assume that the scene contains foreground objects (vehicles, for our autonomous driving application) rigidly moving across a static background. Given an accurate semantic segmentation of some foreground object, the geometry of the pixels within that segment should be spatially and temporally consistent, and the optical flow should be consistent

with the underlying 3D rigid motion.

Due to the high dimensionality of the scene flow problem, we refine our estimates using a cascade of discriminative models [52], with parameters learned via a structural SVM [61]. Every stage of the cascade makes a targeted improvement to one scene variable, implicitly accounting for uncertainty in the current estimates of other scene variables. We initialize our semantic segmentation $\mathcal{S}$ using an instance-level segmentation algorithm [26], and our disparities $\mathcal{D}$ and optical flow fields $\mathcal{F}$ using the PRSF method [134]. We discuss their cascaded refinement next.

4.2.1 Refinement of Semantic Segmentation

The initial single-frame segmentation is unreliable in regions with shadows and reflections. Given stereo inputs, however, our depth estimates provide a strong cue to improve the segmentation. Therefore for each segmentation instance, we define a CRF on the pixels in its enclosing bounding box B_i. We seek to estimate the foreground segmentation s given an initial noisy segmentation $\hat{s}$.

Our data term encourages the inferred segmentation s to be close to the initial segmentation $\hat{s}$. The KITTI scene flow dataset [84] generates "ground truth" segmentations by aligning approximate CAD models, and these annotations are often inaccurate at object boundaries. To add robustness, we thus define a feature $\phi_{\text{dist}}(i, \hat{s})$ by computing the signed distance of pixel i to the original segmentation border, and using a sigmoid function to map these distances to $[0, 1]$. The data energy for our CRF model is then

$$E_{\text{seg}}^{\text{data}}(\mathcal{S}) = \sum_{i \in B_i} \Big[\lambda_1 + \lambda_2 \phi_{\text{dist}}(i, \hat{s})\Big] \delta(s_i = 0, \hat{s}_i = 1) + \Big[\lambda_3 + \lambda_4 \phi_{\text{dist}}(i, \hat{s})\Big] \delta(s_i = 1, \hat{s}_i = 0). \tag{4.1}$$

We demonstrate the benefits of our signed distance feature $\phi_{\text{dist}}(i, \hat{s})$ in Fig. 4.2. By allowing the CRF to reduce confidence in $\hat{s}$ near boundaries, this feature allows other image-based cues to improve segmentation accuracy.

To allow spatial regularization, we add edges $\mathcal{E}$ to our CRF connecting each pixel to its 8 spatial

Signed Distance $\phi_{\text{dist}}(\cdot, \hat{s})$ "Ground Truth" Segmentation CRF Excluding ϕ_{dist} CRF Including ϕ_{dist}

Figure 4.2: The "true" KITTI segmentations [84] are approximate. By incorporating a signed distance feature $\phi_{\text{dist}}(i, \hat{s})$, CRF segmentation accuracy improves.

neighbors:

$$E_{\text{seg}}^{\text{space}}(\mathcal{S}) = \sum_{(i,j)\in\mathcal{E}} \Big[\lambda_5 + \lambda_6 \rho_{\text{img}}(I_i, I_j) + \lambda_7 \rho_{\text{disp}}(d_i, d_j)\Big] \delta(s_i \neq s_j). \tag{4.2}$$

Here, $\rho_{\text{img}}(I_i, I_j) = \exp\{-\frac{||I_i - I_j||}{\sigma_{\text{img}}}\}$ measures RGB color similarity, and $\rho_{\text{disp}}(d_i, d_j) = \exp\{-\frac{|d_i - d_j|}{\sigma_{\text{disp}}}\}$ measures similarity of the current (approximate) disparity estimates.

To learn the parameters $\lambda = [\lambda_1, \ldots, \lambda_7]$, we use a structured SVM [61] with loss equal to the average label error within bounding box B_i [36]. Feature bandwidths $\sigma_{\text{img}}, \sigma_{\text{disp}}$ are tuned using validation data. To perform inference within each iteration of S-SVM learning, we use an efficient message passing implementation of tree-reweighted belief propagation [136, 64]. Because the pixel labels are binary, inference takes less than 0.5s. To apply our CRF model to the scene flow problem, we independently estimate the segmentation for each instance and frame.

4.2.2 Estimation of Scene Geometry

Given a disparity map $\mathcal{D}$ and camera calibration parameters, a 3D point cloud representation of the scene may be constructed. Standard stereo estimation algorithms ignore semantic cues, and often perform poorly on surfaces that are shadowed, reflective, or transparent. As illustrated in Fig. 4.3, for autonomous driving applications the depth estimates for vehicle windshields are especially poor. Because inaccurate depth estimates lead to poor motion and flow estimates, we design a model that enforces local smoothness of depths within inferred segmentation masks.

We define a CRF model of the pixels within each semantic segment previously inferred by

our cascaded model. For each pixel i in the left camera with disparity hypothesis d_i, we denote its corresponding pixel in the right camera as $P_d(i, d_i)$. The data term is defined to penalize the difference in smooth census transform between pixel i and $P_d(i, d_i)$:

$$E_{\text{geom}}^{\text{data}}(\mathcal{D}) = \sum_{\{i|s_i=s\}} \rho_{\text{CSAD}}(I_i, I'_{P_d(i,d_i)}). \tag{4.3}$$

Here, $\rho_{\text{CSAD}}(.,.)$ is the CSAD cost [132] for matched pixels in different images. The CSAD difference is a convex approximation of the census transform [150] that gives reliable pixel correspondences for many datasets [132].

We encourage piecewise-smooth depth maps by penalizing the absolute difference of neighboring pixel depths:

$$E_{\text{geom}}^{\text{space}}(\mathcal{D}) = \tau_1 \sum_{(i,j)\in\mathcal{E}} \rho_{\text{depth}}(d_i, d_j). \tag{4.4}$$

Here $\rho_{\text{depth}}(d_i, d_j) = |\frac{C}{d_i} - \frac{C}{d_j}|$, and C is a camera-specific constant that transforms disparity d into depth $\frac{C}{d}$. We enforce consistency in depth domain because the scale of disparity varies when objects are close or far from the camera.

Although our stereo CRF uses standard features, as illustrated in Fig. 4.3, it is quite effective at resolving uncertainties in challenging regions of foreground objects. If naively applied to the full image, simple CRF models are often inaccurate at object boundaries [133]. However, they are much better able to capture depth variations within a single object. Because our pairwise distances depend only on the absolute value of depth differences, distance transforms [34] may be used for efficient coarse-to-fine inference in the corresponding CRF. On average, it takes less than 5s to perform inference in a 200×200 region with 200 disparity candidates. We refine the disparities for each frame independently.

4.2.3 Estimation of 3D Motion

If the segmentation mask and disparity estimates for each object instance were perfect, we could apply 3D rigid motion to the 3D point cloud for each segment, and project back to the image plane to recover the 2D optical flow. We let (x_i, y_i) denote the *motion flow* constructed in this way. Although

Initial Point Cloud [133] | Initial Disparity Error | Refined Point Cloud | Refined Disparity Error

Figure 4.3: 3D point clouds (top) and corresponding disparity errors (blue small, orange large) for the initial PRSF depth estimates [133], and the refined depth estimates produced by our CRF model.

our imperfect geometry estimates will cause the motion flow to differ from the true optical flow (u_i, v_i), each still provides valuable cues for the estimation of the other.

For each detected segment, we let $M = (R, t)$ denote its 3D relative motion between the first and second frames. The motion M has 6 degress of freedom: t is a translation vector, and $R = (\alpha, \beta, \gamma)$ is a rotation represented by three axis-aligned rotation angles. We match the rigid motion M to the current flow field estimate (u, v) by minimizing the following energy function:

$$E_{\text{motion}}(M) = \nu(\rho(\alpha) + \rho(\beta) + \rho(\gamma)) + \sum_{\{i|s_i=s\}} |x_i(M, d_i) - u_i| + |y_i(M, d_i) - v_i|. \quad (4.5)$$

where $(x_i(M, d_i), y_i(M, d_i))$ is the motion flow computed from disparity d_i, 3D motion M, and the camera calibration. We let $\rho(a) = \sqrt{a^2 + \epsilon^2}$ be the Charbonnier penalty, a smooth function similar to the L_1 penalty that provides effective regularization for motion estimation tasks [124]. We set the regularization constant ν using validation data, and use gradient descent to find the optimal value for M. We visualize an example motion flow map in Fig. 4.4.

4.2.4 Estimation of 2D Optical Flow

The estimated motion flow from the previous stage provides valuable cues for optical flow estimation. As in the example in Fig. 4.4, motion flow errors are primarily caused by imperfect geometries (or disparities). We thus seek a flow field $f_i = (u_i, v_i)$ such that the corresponding pixel $P_f(i, f_i)$ in the next frame matches pixel i, and f_i does not deviate too much from (x_i, y_i). We define a CRF model

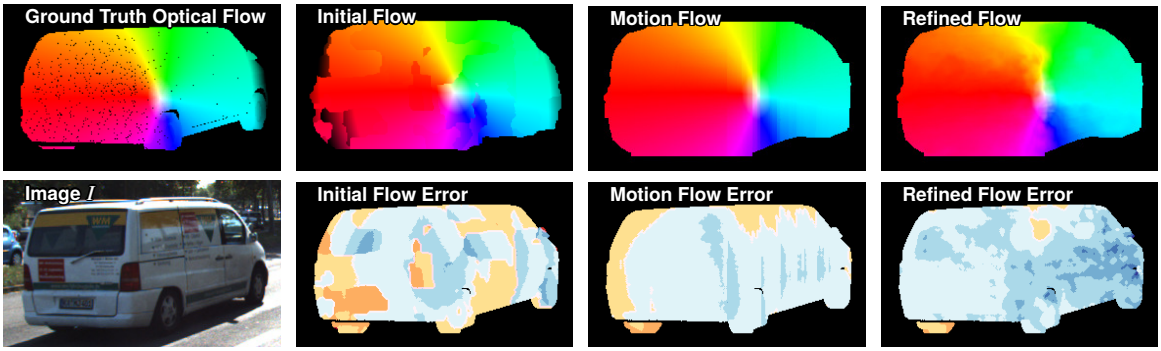


Figure 4.4: Visualization of estimated flow fields (Top, hue encodes orientation [124]) and their error (Bottom, blue small, orange large). A rigid 3D motion flow captures the dominant object motion, and the refined estimates from our CRF model further improve accuracy.

of the pixels within segment s in frame 1, with likelihood

$$E_{\text{flow}}^{\text{data}}(\mathcal{F}) = \sum_{\{i|s_i=s\}} \rho_{\text{CSAD}}(I_i, J_{P_f(i,f_i)}) + \eta_1(|u_i - x_i| + |v_i - y_i|). \tag{4.6}$$

We also encourage spatially smooth flow field estimates:

$$E_{\text{flow}}^{\text{space}}(\mathcal{F}) = \sum_{(i,j)\in\mathcal{E}} \eta_2(|u_i - u_j| + |v_i - v_j|). \tag{4.7}$$

While many optical flow methods use superpixel approximations to make inference more efficient [114], max-product belief propagation can be efficiently implemented for our pixel-level CRF using distance transforms [34, 18]. As shown in Fig. 4.4, our refined optical flow improves the initial flow by smoothly varying across the segment, while simultaneously capturing details that are missed by the motion flow.

To limit the memory consumption of our optical flow algorithm, we perform inference on a down-sampled image and then use the EpicFlow [102] algorithm to interpolate back to the full image resolution. Other recent optical flow algorithms have used a similar approximation [18, 5].

Motion Estimation for Out-of-Frame Pixels We notice that the EpicFlow interpolation tends to produce significant errors for pixels that move outside of the image border. Outside of the camera's field of view, optical flow can only be predicted using the known 3D rigid motion, and we thus

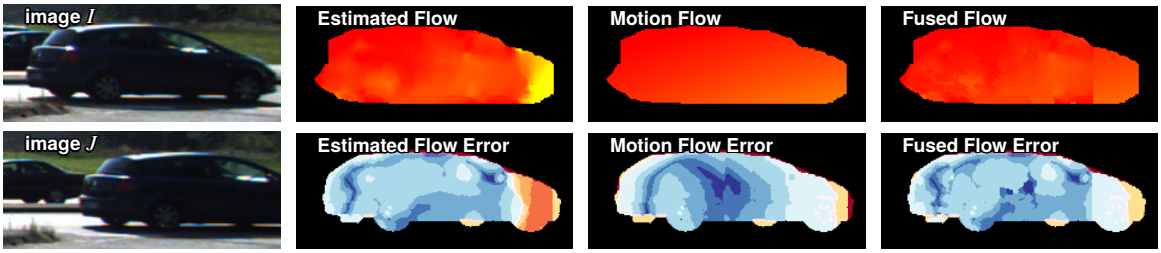


Figure 4.5: Visualization of our flow fusion CRF to reduce motion errors (blue small, orange large) for out-of-border pixels.

propose a flow fusion CRF [73] to combine the estimated optical flow and motion flow for partially occluded objects.

In particular, we use a binary CRF to determine whether the optical flow (u_i, v_i) or motion flow (x_i, y_i) provides a better estimate of the true flow (U_i, V_i) for each pixel i. Intuitively, for within-border pixels we should use the matching cost to compare flow fields, while out-of-border pixels should be biased towards the motion flow interpolation:

$$E^{\text{data}}_{\text{fuse}}(\mathcal{F}) = \omega_1(|U_i - x_i| + |V_i - y_i|)\delta[P_f(i, f_i) \text{ outside}] + \sum_{f=\{(u,v),(x,y)\}} \sum_{\{i|s_i=s\}} \rho_{\text{CSAD}}(I_i, J_{P_f(i,f_i)})\delta[P_f(i, f_i) \text{ inside}].$$

Spatial smoothness is encouraged for neighboring pixels:

$$E^{\text{space}}_{\text{fuse}}(\mathcal{F}) = \sum_{(i,j)\in\mathcal{E}} \omega_2(|U_i - U_j| + |V_i - V_j|) \tag{4.8}$$

We tune parameters ω_1, ω_2 using validation data, and minimize the energy using tree-reweighted belief propagation [64]. We show in Fig. 4.5 that the fused flow estimate retains many details of the optical flow, while using the motion flow to better interpolate in occluded regions. We also apply our flow fusion technique to update the noisy background flow predictions. See Fig. 4.9 for additional examples of our final optical flow estimates.

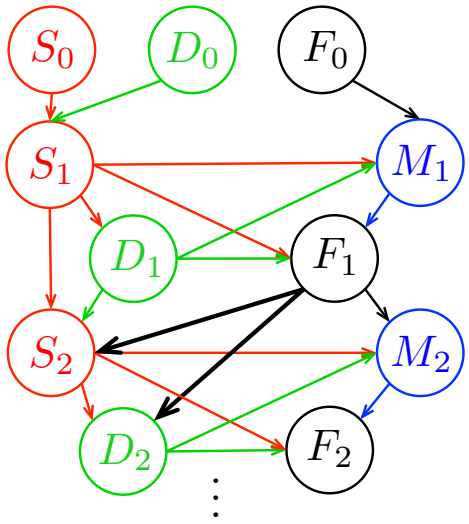


Figure 4.6: A directed graph summarizing our cascaded approach to the estimation of object segmentations S, disparities D, 3D rigid motions M, and optical flow F. Subscripts indicate different stages of the cascade. The bold arrows represent additional temporal dependencies added for stages two and later.

4.3 Cascaded Scene Flow Prediction

The CRF models defined in Sec. 4.2 refine the various components of our scene model greedily, by estimating each one given the current best estimates for all others. However, this approach does not fully utilize the temporal relationships between the segmentation and geometry at different frames. Also, when the initial optical flow contains major errors, our motion flow estimates will be inaccurate. To better capture the full set of geometric and temporal relationships, we thus use multiple stages of cascaded prediction [52] to further refine our scene flow estimates. The inputs and outputs for each stage of our cascade are summarized by the directed graph in Fig. 4.6.

Temporal Segmentation Consistency Rather than segmenting each video frame independently, in the second stage of our cascade, we use the inferred flow field f to encourage temporal consistency.

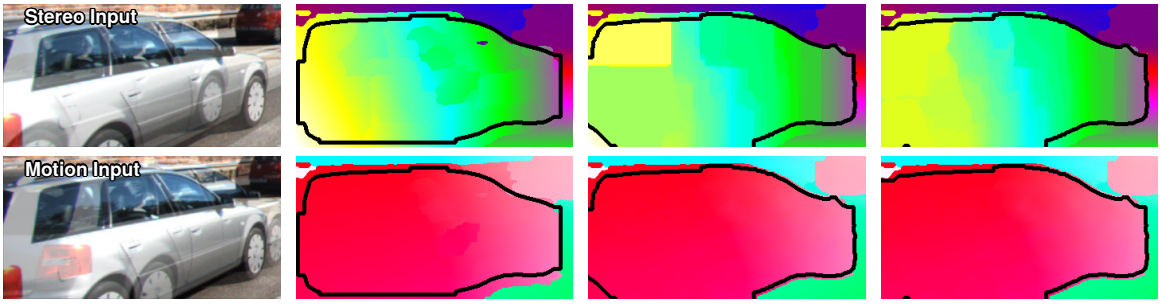


Figure 4.7: From top to bottom, we visualize input frames, initial disparity (left) and flow (right) predictions, and the refined disparity and flow after the first and second stages of the cascade. Our refined flow estimates from stage 1 (note object boundaries) lead to improved stereo estimates in stage 2 (upper left).

Each pixel i in frame 1 is linked to matched pixel $P_f(i, f_i)$ in frame 2:

$$E^{\text{time}}_{\text{seg}}(\mathcal{S}) = \lambda_8 \delta(s^{(1)}_i = 0, s^{(2)}_{P_f(i,f_i)} = 1) + \lambda_9 \delta(s^{(1)}_i = 1, s^{(2)}_{P_f(i,f_i)} = 0) + \sum_i \left[\lambda_{10} + \lambda_{11} \rho_{\text{CSAD}}(I_i, J_{P_f(i,f_i)})\right] \delta(s^{(1)}_i = s^{(2)}_{P_f(i,f_i)}).$$

We again use S-SVM learning of CRF parameters λ, and infer segmentations using tree-reweighted belief propagation.

Temporal Geometric Consistency As in our temporal segmentation model, we also extend the stereo CRF of Sec. 4.2.2 to encourage smooth changes for the depths of pixels linked by our optical flow estimates:

$$E^{\text{time}}_{\text{geom}}(\mathcal{D}) = \tau_2 \sum_{\{i | s^{(1)}_i = s\}} \rho_{\text{depth}}(d_i(m_i), d_{P_f(i,f_i)}). \tag{4.9}$$

Here, $d_i(m_i)$ denotes the disparity value of pixel i in the second frame when rigid motion m_i is applied. The parameters τ are learned using validation data. Fig. 4.7 shows an example of the improved disparity and flow estimates produced across multiple stages of our cascade.

Recovery from a Poor Optical Flow Initialization If the initial noisy optical flow is very inaccurate, our cascade cannot recover the correct 3D motions of objects because we assume motion flow should match optical flow. Since our updated semantic segmentation masks $s^{(1)}$ and $s^{(2)}$ are

typically very accurate, when applying rigid motion M to pixels in $s^{(1)}$, the shape of the new segmentation mask $s(M)$ should be similar to $s^{(2)}$. We measure this similarity via a cost defined on the second-frame bounding box B:

$$\frac{1}{|B|}\sum_{i\in B}\alpha S(M)_i \cdot C(S_i^{(2)}) + (1-\alpha)C(S(M)_i)\cdot S_i^{(2)}. \tag{4.10}$$

Here, $C(\cdot)$ is the Chamfer difference map and $\alpha = 0.5$. This cost function is widely used for human pose estimation [8]. If this cost exceeds 0.5, we replace the first term in Eq. (4.5) with this silhouette cost. By optimizing this modified objective, we can recover from bad motion estimates.

To provide an intuitive example, we visualized an example in Figure 4.8. Since the initial flow is totally wrong, the estimated motion will inevitably be problematic as well, and indeed the silhouette cost is large. For flow estimations with large silhouette errors, we replace the first term in Eq. (5) with this term. As a result, the estimated motion is reliable, leading to more accurate optical flow predictions.

Second Frame Disparities For the KITTI scene flow dataset [84], the ground truth disparity for the second frame is represented as per-pixel disparity changes with respect to the first frame. To predict this quantity for evaluation, we apply our estimated 3D rigid motion for each pixel to its estimated geometry in the first frame. The accuracy of these disparity estimates is thus strongly dependent on the performance of our motion estimation algorithm.

Figure 4.8: Visualizing how we recover correct 3D motion from poor flow initializations by minimizing the silhouette cost.

	D1-bg	D1-fg	D1-all	D2-bg	D2-fg	D2-all	Fl-bg	Fl-fg	Fl-all	SF-bg	SF-fg	SF-all	Time
PRSF	4.74	13.74	6.24	11.14	20.47	12.69	11.73	27.73	14.39	13.49	31.22	16.44	2.5min
CSF	4.57	13.04	5.98	7.92	20.76	10.06	10.40	30.33	13.71	12.21	33.21	15.71	**1.3min**
OSF	4.54	12.03	5.79	5.45	19.41	7.77	5.62	22.17	8.37	7.01	26.34	10.23	50min
SSF-O	4.30	**8.72**	5.03	5.13	**15.27**	**6.82**	**5.42**	17.24	7.39	**6.95**	25.78	10.08	52.5min
SSF-P	**3.55**	8.75	**4.42**	**4.94**	17.48	7.02	5.63	**14.71**	**7.14**	7.18	**24.58**	**10.07**	5min

Table 4.1: Scene flow results on all pixels for KITTI test set. Under most evaluation metrics, our algorithm SSF outperforms PRSF [133], CSF [80], OSF [84] that take two frames as input.

	D1-bg	D1-fg	D1-all	D2-bg	D2-fg	D2-all	Fl-bg	Fl-fg	Fl-all	SF-bg	SF-fg	SF-all
CSF	4.03	11.82	5.32	6.39	16.75	8.25	8.72	26.98	12.03	10.26	28.68	13.56
PRSF	4.41	13.09	5.84	6.35	16.12	8.10	6.94	23.64	9.97	8.35	26.08	11.53
OSF	4.14	11.12	5.29	4.49	16.33	6.61	4.21	18.65	6.83	5.52	22.31	8.52
SSF-O	3.98	7.82	4.62	4.26	**12.31**	**5.70**	**4.04**	13.18	5.70	**5.44**	21.11	**8.25**
SSF-P	**3.30**	**7.74**	**4.03**	**4.12**	14.57	5.99	4.20	**10.81**	**5.40**	5.70	**19.93**	**8.25**

Table 4.2: Scene flow results on non-occluded pixels for KITTI test set. SSF also outperforms other methods.

4.4 Experiment

We test our semantic scene flow algorithm (**SSF**) with 3 iterations of cascaded prediction on the challenging KITTI 2015 benchmark [84]. The performance is evaluated at disparity estimations at two frames (**D1**, **D2**), flow estimations (**Fl**) at the reference frame, and scene flow estimations (**SF**), and evaluated separately on foreground (**fg**), background (**bg**) and all pixels (**all**). See Table 4.1 for experimental results at all pixels and Table 4.2 for non-occluded pixels. We test SSF with PRSF [133] initialization (**SSF-P**) and OSF [84] initialization (**SSF-O**). Our algorithm is superior to state-of-the-art scene flow algorithms in all the evaluation metrics and we also provide qualitative results on training sets in Figure 4.9.

In Table 4.3, we test the performance gain in each stage of the cascade in the training set. There is an improvement at the first iteration of the cascade when modeling segmentation and geometry at independent frames, followed by another improvement at the second iteration when temporal consistency is introduced. At the third iteration, the performance starts to saturate.

Speed Scene flow prediction usually takes a long time to compute, and efficient algorithms [80] usually trade efficiency over accuracy. Training on a large synthetic dataset, CNN-based approaches

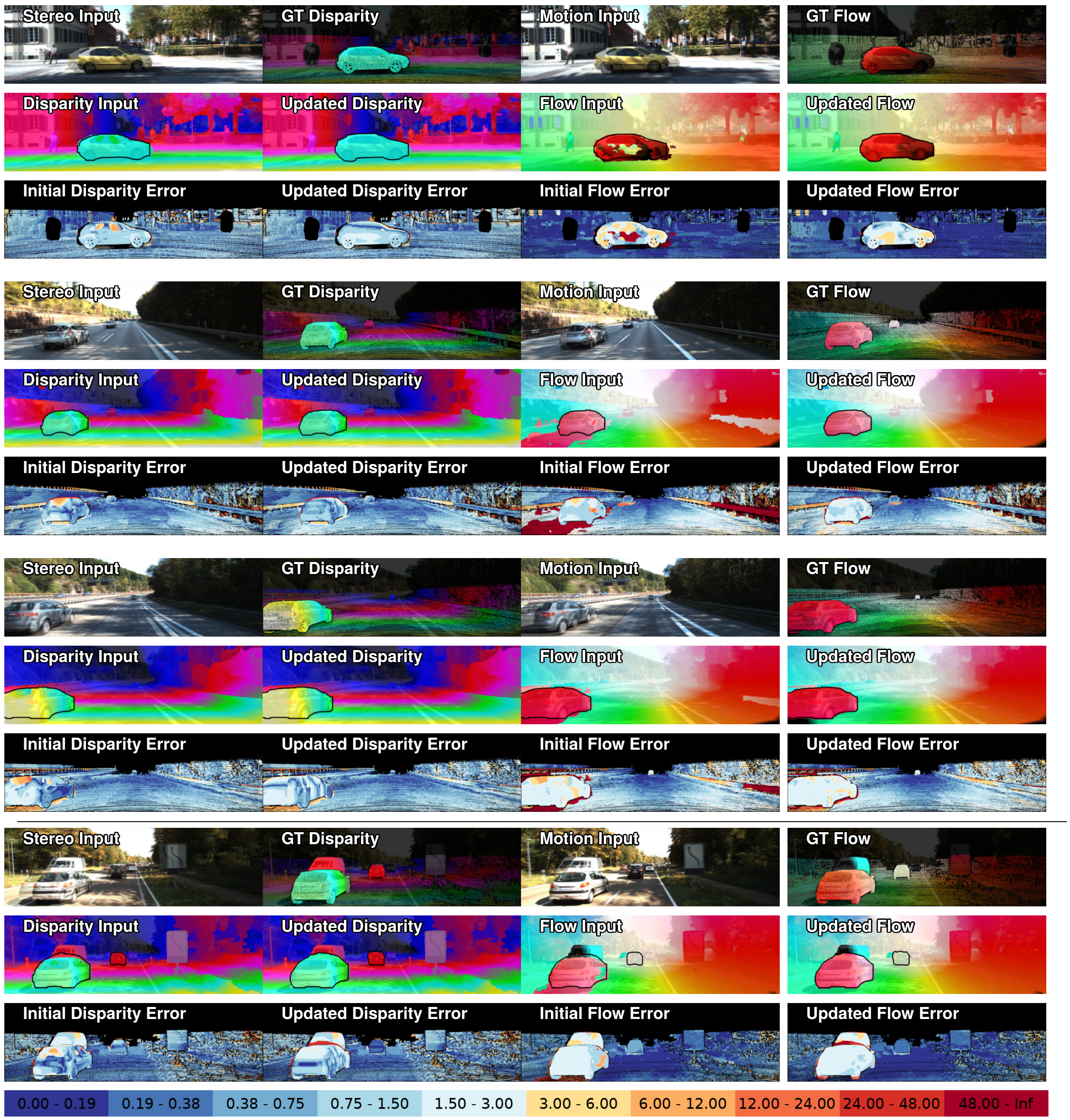


Figure 4.9: Visualizing the qualitative results of our algorithm for 4 sequences in KITTI training set. In the last set of results, we show one failure case where the imperfect segmentations lead to scene flow estimation error.

	Seg	D1-fg	D1-bg	Fl-fg	Fl-bg
PRSF	10.1	5.00	3.27	8.54	4.04
Iter 1	9.79	3.69	3.18	8.38	3.67
Iter 2	8.41	3.50	3.15	8.20	3.65
Iter 3	8.40	3.49	3.15	8.20	3.65
GT Seg	0	2.19	3.05	7.61	3.56

Table 4.3: Results on KITTI validation set. Starting with noisy PRSF initialization, we make improvements at each stage of the cascade. In the last row, we show that when segmentation mask is perfect, scene flow prediction can be improved in a huge margin.

have been proposed to perform optical flow [30, 140] and disparity estimation [83] as two separate tasks at very fast speeds. However, their accuracy is not as good as classical scene flow algorithms on the real-world KITTI dataset. Although the parameter space of our algorithm is huge and we are making pixel level predictions, our algorithm is still efficient. The main reason is that we disentangle the output space, and utilize efficient algorithms [34, 18] to solve each inference problem with high dimensional outputs. Most of the computation time is spent on feature computation, and could be improved using parallel computation.

Failure Cases As can be seen in Figure 4.9, if the instance segmentation algorithm fails to detect missing vehicles for challenging cases, the performance of our algorithm can be influenced. Similar to optical flow methods that utilize semantic information [5], we conducted an experiment using ground truth segmentation masks and witnessed a significant performance gain, see Table 4.3. Therefore our bottleneck is the accuracy of instance segmentation.

4.5 Conclusion

In this work, we utilize semantic cues to identify rigidly moving objects, and thereby produce more accurate scene flow estimates for real-world scenes. Our cascaded prediction framework allows computationally efficient recovery of high-dimensional motion and geometry estimates, and can flexibly utilize cues from sophisticated semantic segmentation algorithms. We improve on the state-of-the-art for the challenging KITTI scene flow benchmark [84, 40]. While our experiments have focused on using vehicle detections to improve scene flow estimates for autonomous driving, our

cascaded scene flow method is directly applicable to any category of objects with near-rigid motion.

4.6 Supplementary Material

4.6.1 Learned Parameters

We split ground truth data for KITTI dataset [84] as 150 for training and 50 for validation, and learn parameters in our model at each stages of the cascade. Here, we report the learned parameters.

Refinement for Semantic Segmentation

We show in Table 4.4 the learned parameters for segmentation refinement model using Structural SVM training [36].

Iter1	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
	-1.90	3.68	1.62	-3.29	0.00	0.64	0.17
	λ_8	λ_9	λ_{10}	λ_{11}	σ_{img}	σ_{disp}	
	-	-	-	-	100	3000	
Iter2	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
	-0.45	0.92	0.36	-0.69	0.04	0.12	-0.03
	λ_8	λ_9	λ_{10}	λ_{11}	σ_{img}	σ_{disp}	
	-0.03	0.08	0.02	0.01	100	3000	
Iter3	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
	-0.10	0.25	0.09	-0.11	0.03	0.03	-0.01
	λ_8	λ_9	λ_{10}	λ_{11}	σ_{img}	σ_{disp}	
	0.09	-0.00	0.02	100	5000	5000	

Table 4.4: Parameters ($\times 10^{-3}$) for segmentation refinement.

Estimating Scene Geometry, 3D Motion, 2D Optical Flow, and Flow Fusion

We learn the rest of the parameters for scene geometry modeling, 3D motion and 2D optical flow estimation, and flow fusion using grid search. The learned parameters will be shared across all stages of the cascade and is shown in Table 4.5.

Scene Geometry Estimation	τ_1	τ_2
	3	0.5
3D Motion Estimation	ν	
	10	

Optical Flow Estimation	η_1	η_2
	1	0.12
Flow Fusion	ω_1	ω_2
	1	0.1

Table 4.5: Parameters for estimating scene geometry, 3D Motion, 2D optical flow, and flow fusion.

4.6.2 More Qualitative Results

We demonstrate more qualitative results of our algorithm on KITTI training set [84] in Figure 4.10, 4.11 and 4.12.

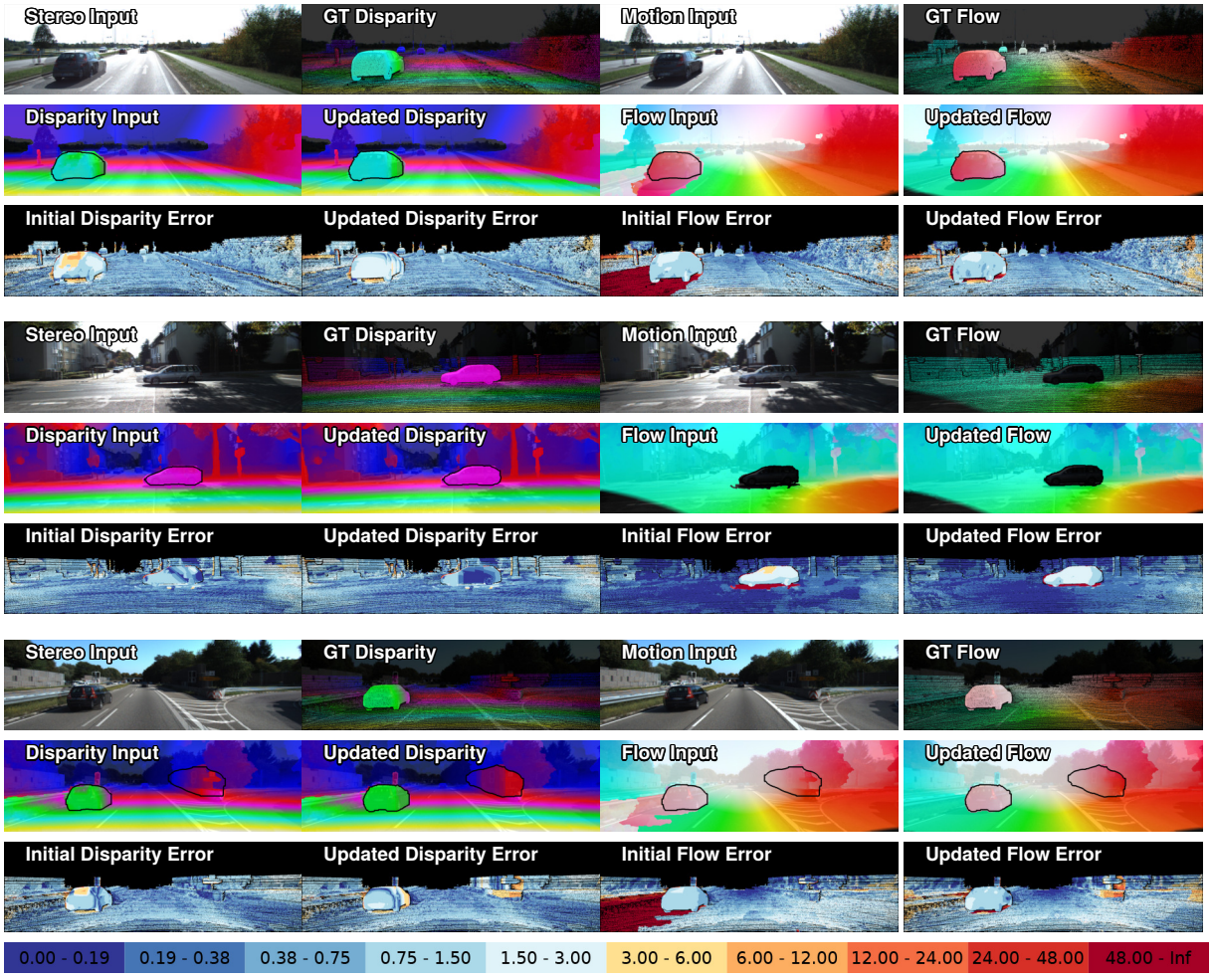


Figure 4.10: Visualizing the qualitative results of our algorithm in KITTI training set.

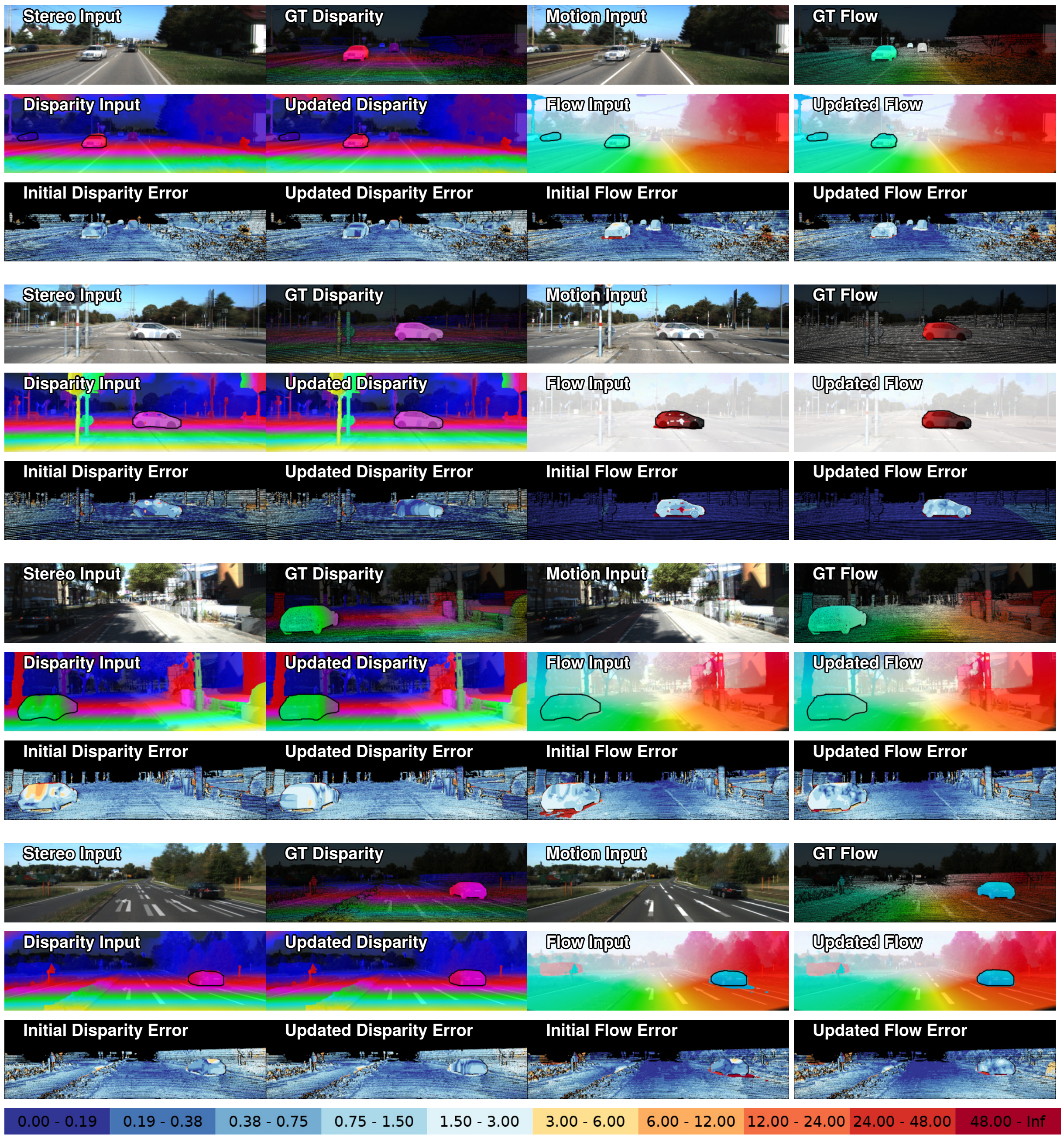


Figure 4.11: Visualizing the qualitative results of our algorithm in KITTI training set. (contd.)

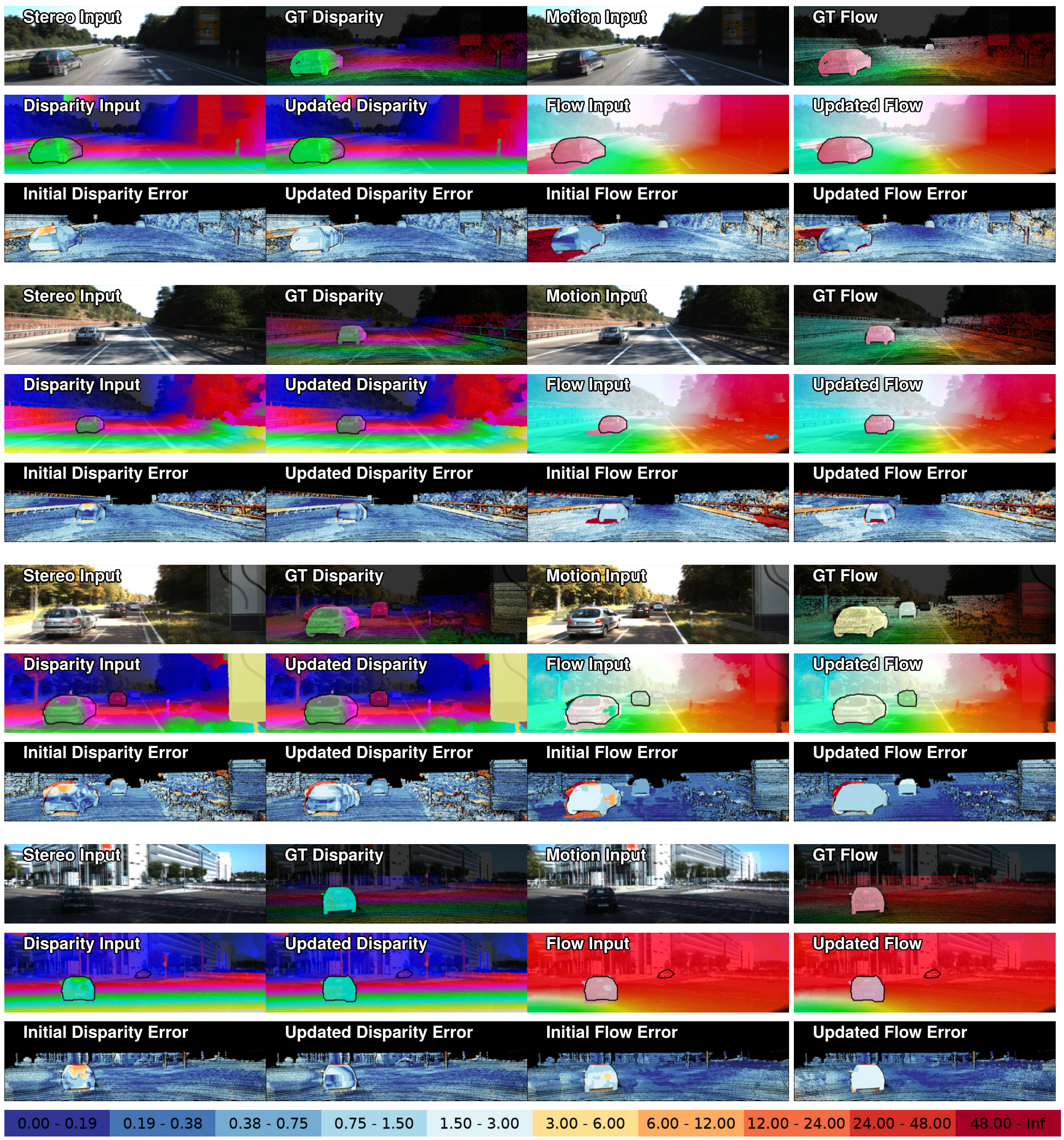


Figure 4.12: Visualizing the qualitative results of our algorithm in KITTI training set. (contd.)

Chapter 5

Future Research

In this thesis, we developed new representations and algorithms for 3D scene understanding. Our *clouds of oriented gradient* feature is an orientation-invariant 3D appearance descriptor. By modeling *latent support surfaces*, our model is able to capture style variations and help detect small objects. Via structured training and cascaded prediction, our algorithm is able to learn contextual relationship among objects and achieve state-of-the-art performance in 3D detection tasks [99, 100]. We also use cascaded prediction to solve outdoor scene flow prediction tasks by jointly modeling semantic segmentation, scene geometry, and motion [101]. We now conclude by discussing open research directions in 3D scene understanding and motion estimation.

Over the last few decades, we witnessed the evolution of 2D object detection systems (Fig. 5.1). Early works used hand-crafted features [27] to model object apperance, and were improved by using a part-based descriptor [33]. In modern computer vision systems, convolutional neural networks (CNNs) have become the most widely used method to extract rich features from images. One reason why CNN-based systems are effective is because networks are trained using large-scale datasets like ImageNet [103].

Our thesis work on 3D object detection also follows the evolution of 2D detection systems (Fig. 5.1). The clouds of oriented gradient feature is a hand-crafted 3D appearance feature, and the latent support surfaces can be viewed as a part-based 3D descriptor. 3D object detection has a relatively shorter history because the first accurately labelled benchmark [118] was just introduced

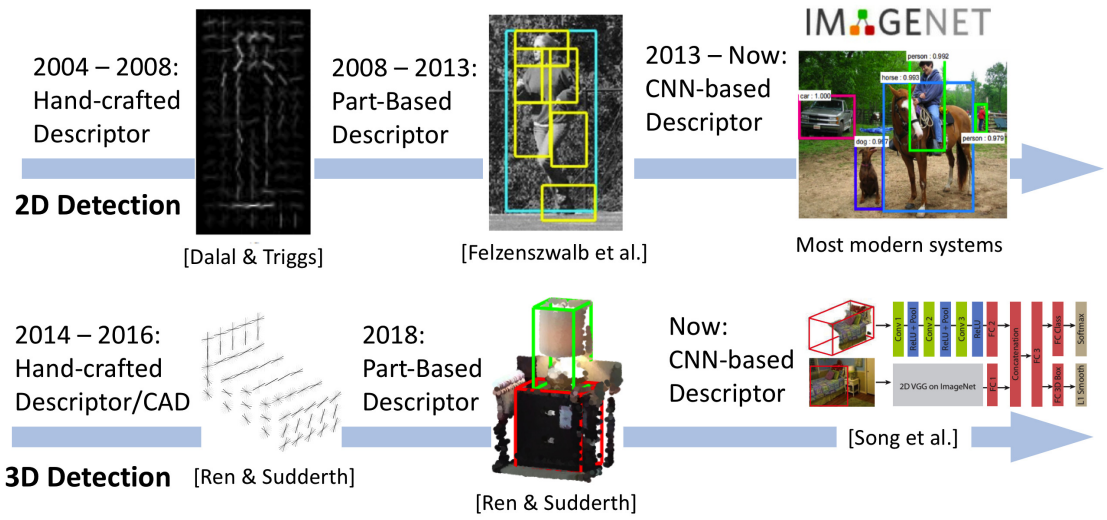


Figure 5.1: The evolution of 2D and 3D object detection systems. Our thesis work on 3D detection [99, 100] lines up with the advancement of 2D detection systems. Designing 3D CNN-based object detection is a tempting idea, but it is challenging without large scale datasets with accurate 3D annotations.

in 2015. Recently, people also started to train 3D CNN-based systems. However as we described in Chapter 3, our approaches that use COG and cascaded prediction are able to achieve compatible or even better performance than CNN-based systems [120, 69]. This suggests that directly applying 3D convolutional networks is less effective. One major reason is that labeling 3D datasets is very time-consuming, and no real-world 3D dataset has ImageNet's scale. As a result, learned convolutional features from this relatively small dataset may not generalize well. There are also a few recent works that convert 2D bounding box proposals to 3D cuboids [90, 29], but their systems heavily rely on state-of-the-art 2D detection systems [42] trained on external 2D datasets. A similar issue also happens in motion estimation tasks, where real-world datasets are very small because constructing ground truth motion fields is very difficult.

With advanced rendering technologies, synthetically simulated environments are closer to realistic scenes, and they are becoming informative benchmarks for various scene understanding tasks. One major benefit of synthetically generated environments is that their annotations are accurate and scalable. Since 2D CNN systems for 2D scene understanding tasks have benefited from large-scale datasets, we consider the use of synthetic 3D datasets for solving 3D scene understanding tasks. We

argue that it is a promising direction for designing computer vision systems.

Photo-Realistic Synthetic Datasets for 3D CNNs There are a few existing works that use graphic engines to create 3D datasets. For example, the SUN CG dataset [122] was rendered by physically-based graphic engines [153] to provide accurate annotations in indoor scenes. However the level of photo-realism is low, thus the features trained from this dataset may not generalize well to real-world datasets. The rendered scenes in AI2-THOR dataset [65] are closer to realism, but the size of the dataset is small.

Recent CNN-based 3D object detection systems [90, 69, 29] first use 2D detection systems [97] to initialize bounding box proposals, then a deep neural network structure is used to recover 3D bounding boxes. Such systems are not learning representations solely from 3D data, but rather rely on advanced 2D deep learning systems. This is because no large 3D dataset is available for deep learning systems to learn rich descriptors.

We believe it is promising to use more 3D meshes and more complex lighting settings to generate photo-realistic datasets with cuboid annotations (Fig. 5.2). Since the size of the synthetic dataset can be very and annotations are accurate, learned 3D convolutional features are likely to be cleaner and useful. The photo-realism of the dataset may allow learned features to generalize well to real-world data. Therefore with such a dataset, training 3D CNN-based detection systems is likely to be more effective.

Figure 5.2: Photo-realistic synthetic 3D environments provide accurate annotations at large scale.

Large-scale Synthetic Datasets for Motion Estimation For motion estimation tasks such as optical flow or scene flow, modern motion estimation systems are also starting to use synthetic datasets to train deep neural networks (Fig. 5.3).

Despite recent advances in optical flow estimation, it is still challenging to account for complicated motion patterns. At video rates, however, even such complicated motion patterns are smooth for longer than just two consecutive frames. This suggests that information from frames that are adjacent in time could be used to improve the optical flow estimation.

Indeed numerous methods have been developed to either impose temporal smoothness of the flow [11, 12] or to explicitly reason about the trajectory of each pixel across multiple frames [107]. Despite the fact that multiple frames carry additional information, none of the top three optical flow algorithms on the major benchmark datasets uses more than two frames [15, 40].

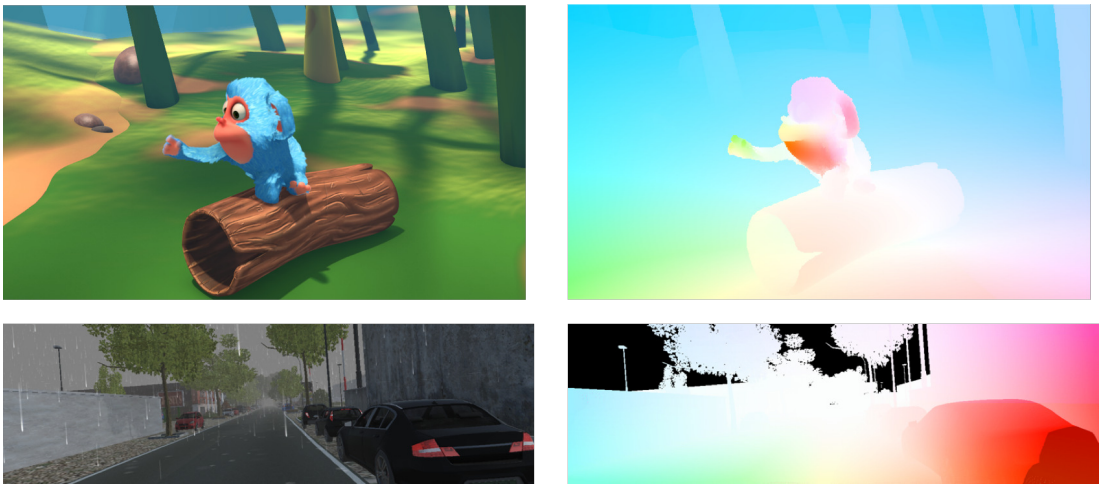


Figure 5.3: Large-scale synthetically generated datasets like Monkaa [83] (top) and virtual KITTI [39] (bottom) can be used to train deep neural networks for motion estimation tasks.

This may be due to the fact that motion and its statistics do change over time. If the optical flow field changes dramatically the visual information contained in longer frame sequences may be less useful and potentially detrimental [135]. The ability to decide when visual information from past frames is useful is paramount to the success of multi-frame optical flow algorithms. Some early methods account for the temporal dynamics of motion using a Kalman filter [31, 22].

More recent appraches attenuate the temporal smoothness requirement when a sudden change is detected [106, 135].

Training on large scale synthetic dataset [83], CNN-based optical flow estimation algorithms [126, 58] already perform on par with variational approaches [145]. Those synthetic datasets also contain potentials for deep networks, such as RNN [23], to learn temporal information. A promising research direction is to design a multi-frame scene flow network that accounts for geometric information and 3D motion. Training on large scale synthetic datasets, systems with this network structure are likely to achieve improved performance on real-world benchmarks.

Bibliography

[1] SUNw: Scene understanding workshop. http://sunw.csail.mit.edu/, 2017.

[2] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1534–1543, 2016.

[3] Mathieu Aubry, Daniel Maturana, Alexei Efros, Bryan Russell, and Josef Sivic. Seeing 3D chairs: Exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[4] Junjie Bai, Qi Song, Olga Veksler, and Xiaodong Wu. Fast dynamic programming for labeling problems with ordering constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1728–1735. IEEE, 2012.

[5] Min Bai, Wenjie Luo, Kaustav Kundu, and Raquel Urtasun. Exploiting semantic information and deep matching for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 154–170. Springer, 2016.

[6] Christian Bailer, Kiran Varanasi, and Didier Stricker. Cnn-based patch matching for optical flow with thresholded hinge loss. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

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

[8] Alexandru O Balan, Leonid Sigal, Michael J Black, James E Davis, and Horst W Haussecker. Detailed human shape and pose from images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

[9] Aayush Bansal, Bryan Russell, and Abhinav Gupta. Marr revisited: 2D-3D alignment via surface normal prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5965–5974, 2016.

[10] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios? In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[11] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1991.

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

[13] Thomas Brox and Jitendra Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2011.

[14] Norbert Buch, James Orwell, and Sergio A Velastin. 3D extended histogram of oriented gradients (3dhog) for classification of road users in urban scenes. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

[15] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 611–625. Springer, 2012.

[16] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[17] Chenyi Chen, Ming-Yu Liu, Oncel Tuzel, and Jianxiong Xiao. R-CNN for small object detection. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 214–230. Springer, 2016.

[18] Qifeng Chen and Vladlen Koltun. Full flow: Optical flow estimation by global optimization over regular grids. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3D object proposals for accurate object class detection. In *Advances in Neural Information Processing Systems (NIPS)*, pages 424–432, 2015.

[20] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3D object proposals using stereo imagery for accurate object class detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[21] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] Toshio M Chin, William Clement Karl, and Alan S Willsky. Probabilistic and sequential computation of optical flow using temporal coherence. *IEEE Transactions on Image Processing (TIP)*, 1994.

[23] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[24] James M Coughlan and Alan L Yuille. Manhattan world: Compass direction from a single image by Bayesian inference. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 941–947. IEEE, 1999.

[25] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[27] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005.

[28] Andrew J Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[29] Zhuo Deng and Longin Jan Latecki. Amodal detection of 3D objects: Inferring 3D bounding boxes from 2d ones in rgb-depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[30] Alexey Dosovitskiy, Philipp Fischery, Eddy Ilg, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[31] Michael Elad and Arie Feuer. Recursive optical flow estimationâĂŤadaptive filtering approach. *Journal of Visual Communication and Image Representation*, 1998.

[32] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[33] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(9):1627–1645, 2010.

[34] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision (IJCV)*, 70(1):41–54, 2006.

[35] Sanja Fidler, Sven Dickinson, and Raquel Urtasun. 3D object detection and viewpoint estimation with a deformable 3D cuboid model. In *Advances in Neural Information Processing Systems (NIPS)*, pages 611–619, 2012.

[36] Thomas Finley and Thorsten Joachims. Training structural svms when exact inference is intractable. In *International Conference on Machine Learning (ICML)*. ACM, 2008.

[37] David Ford Fouhey, Abhinav Gupta, and Martial Hebert. Unfolding an indoor origami world. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 687–702. Springer, 2014.

[38] Yasutaka Furukawa and Jean Ponce. Dense 3D motion capture from synchronized video streams. In *Image and Geometry Processing for 3-D Cinematography*. Springer, 2010.

[39] A Gaidon, Q Wang, Y Cabon, and E Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[40] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012.

[41] Andreas Geiger and Chaohui Wang. Joint 3D object and layout inference from a single RGB-D image. In *German Conference on Pattern Recognition (GCPR)*, 2015.

[42] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[43] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[44] Fatma Guney and Andreas Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4165–4175, 2015.

[45] Ruiqi Guo and Derek Hoiem. Support surface prediction in indoor scenes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2144–2151. IEEE, 2013.

[46] Abhinav Gupta, Alexei A Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 482–496. Springer, 2010.

[47] Saurabh Gupta, Pablo Andrés Arbeláez, Ross B. Girshick, and Jitendra Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[48] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 345–360. Springer, 2014.

[49] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[50] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1849–1856. IEEE, 2009.

[51] Varsha Hedau, Derek Hoiem, and David Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 224–237. Springer, 2010.

[52] Geremy Heitz, Stephen Gould, Ashutosh Saxena, and Daphne Koller. Cascaded classification models: Combining models for holistic scene understanding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 641–648, 2009.

[53] Derek Hoiem, Alexei Efros, Martial Hebert, et al. Geometric context from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 654–661. IEEE, 2005.

[54] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 1981.

[55] Peiyun Hu and Deva Ramanan. Finding tiny faces. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[56] Frédéric Huguet and Frédéric Devernay. A variational method for scene flow estimation from stereo sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

[57] Junhwa Hur and Stefan Roth. Joint optical flow and temporally consistent semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016.

[58] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[59] Zhaoyin Jia, Andrew Gallagher, Ashutosh Saxena, and Tsuhan Chen. 3D-based reasoning with blocks, support, and stability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2013.

[60] Hao Jiang and Jianxiong Xiao. A linear approach to matching cuboids in RGBD images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[61] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural svms. *Machine Learning*, 2009.

[62] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 21(5):433–449, 1999.

[63] Byung-soo Kim, Pushmeet Kohli Kohli, and Silvio Savarese. 3D scene understanding by voxel-CRF. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[64] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(10), 2006.

[65] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[66] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. 2011.

[67] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289. Morgan Kaufmann, 2001.

[68] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on Graphics (SIGGRAPH)*, 33(4):149, 2014.

[69] Jean Lahoud and Bernard Ghanem. 2D-driven 3D object detection in RGB-D images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[70] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1817–1824. IEEE, 2011.

[71] Chen-Yu Lee, Vijay Badrinarayanan, Tomasz Malisiewicz, and Andrew Rabinovich. Roomnet: End-to-end room layout estimation. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[72] Daniel C Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2136–2143. IEEE, 2009.

[73] Victor Lempitsky, Stefan Roth, and Carsten Rother. Fusionflow: Discrete-continuous optimization for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[74] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5325–5334, 2015.

[75] Joseph J Lim, Aditya Khosla, and Antonio Torralba. FPM: Fine pose parts-based model with 3D CAD models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 478–493. Springer, 2014.

[76] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing IKEA objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[77] Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3D object detection with RGBD cameras. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1417–1424. IEEE, 2013.

[78] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr DollÃąr. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[79] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5695–5703, 2016.

[80] Zhaoyang Lv, Chris Beall, Pablo F Alcantarilla, Fuxin Li, Zsolt Kira, and Frank Dellaert. A continuous optimization approach for efficient and accurate scene flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016.

[81] Arun Mallya and Svetlana Lazebnik. Learning informative edge maps for indoor scene layout prediction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 936–944, 2015.

[82] Daniel Maturana and Sebastian Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 922–928. IEEE, 2015.

[83] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[84] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3061–3070, 2015.

[85] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. 3D bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5632–5640. IEEE, 2017.

[86] Thomas Müller, Jens Rannacher, Clemens Rabe, and Uwe Franke. Feature-and depth-supported modified total variation optical flow for 3D motion field estimation in real scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[87] Sebastian Nowozin and Christoph H Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3–4):185–365, 2011.

[88] Luca Del Pero, Jinyan Guan, Ernesto Brau, Joseph Schlecht, and Kobus Barnard. Sampling bedrooms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2009–2016. IEEE, 2011.

[89] Erik Learned-Miller Pia Bideau. It's moving! a probabilistic model for causal motion segmentation in moving camera videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[90] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[91] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[92] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[93] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

[94] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[95] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[96] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[97] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[98] Zhile Ren and Gregory Shakhnarovich. Image segmentation by cascaded region agglomeration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2011–2018, 2013.

[99] Zhile Ren and Erik B Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1525–1533, 2016.

[100] Zhile Ren and Erik B Sudderth. 3D object detection with latent support surfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 937–946, 2018.

[101] Zhile Ren, Deqing Sun, Jan Kautz, and Erik Sudderth. Cascaded scene flow prediction using semantic segmentation. In *International Conference on 3D Vision (3DV)*, pages 225–233. IEEE, 2017.

[102] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[103] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[104] Bryan C Russell and Antonio Torralba. Building a database of 3D scenes from user annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2711–2718. IEEE, 2009.

[105] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[106] Agustín Salgado and Javier Sánchez. Temporal constraints in large optical flow estimation.

[107] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. *International Journal of Computer Vision (IJCV)*, 80(1):72–91, October 2008.

[108] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47(1-3):7–42, 2002.

[109] Maximilian Scherer, Michael Walter, and Tobias Schreck. Histograms of oriented gradients for 3D object retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010.

[110] Alexander G Schwing, Tamir Hazan, Marc Pollefeys, and Raquel Urtasun. Efficient structured prediction for 3D indoor scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2815–2822. IEEE, 2012.

[111] Alexander Gerhard Schwing, Sanja Fidler, Marc Pollefeys, and Raquel Urtasun. Box in the box: Joint 3D layout and object reasoning from single images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 353–360. IEEE, 2013.

[112] Akihito Seki and Marc Pollefeys. Sgm-nets: Semi-global matching with neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–26, 2017.

[113] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[114] Laura Sevilla-Lara, Deqing Sun, Varun Jampani, and Michael J Black. Optical flow with semantic segmentation and localized layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[115] Tianjia Shao, Aron Monszpart, Youyi Zheng, Bongjin Koo, Weiwei Xu, Kun Zhou, and Niloy J Mitra. Imagining the unseen: Stability-based cuboid arrangements for scene understanding. *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 33(6), 2014.

[116] Yichang Shih, Sylvain Paris, Frédo Durand, and William T Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 32(6):200, 2013.

[117] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 746–760. Springer, 2012.

[118] Shuran Song, Lichtenberg Samuel, and Jianxiong Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015.

[119] Shuran Song and Jianxiong Xiao. Sliding shapes for 3D object detection in depth images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 634–651. Springer, 2014.

[120] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[121] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[122] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[123] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[124] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010.

[125] Deqing Sun, Stefan Roth, and Michael J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision (IJCV)*, 2014.

[126] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[127] Tatsunori Taniai, Sudipta N. Sinha, and Yoichi Sato. Fast multi-frame stereo scene flow with motion segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[128] Shubham Tulsiani, Abhishek Kar, João Carreira, and Jitendra Malik. Learning category-specific deformable 3D models for object reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2016.

[129] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial occulsion. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[130] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1999.

[131] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–I. IEEE, 2001.

[132] Christoph Vogel, Stefan Roth, and Konrad Schindler. An evaluation of data costs for optical flow. In *German Conference on Pattern Recognition (GCPR)*. Springer, 2013.

[133] Christoph Vogel, Konrad Schindler, and Stefan Roth. Piecewise rigid scene flow. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1377–1384, 2013.

[134] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision (IJCV)*, 115(1), 2015.

[135] Sebastian Volz, Andres Bruhn, Levi Valgaerts, and Henning Zimmer. Modeling temporal coherence for optical flow. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[136] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE transactions on information theory*, 51(11):3697–3717, 2005.

[137] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[138] Andreas Wedel, Thomas Brox, Tobi Vaudrey, Clemens Rabe, Uwe Franke, and Daniel Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision (IJCV)*, 95(1):29–51, 2011.

[139] Andreas Wedel, Clemens Rabe, Tobi Vaudrey, Thomas Brox, Uwe Franke, and Daniel Cremers. Efficient dense scene flow from sparse or dense stereo data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 739–751. Springer, 2008.

[140] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[141] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 3D shapenets for 2.5D object recognition and next-best-view prediction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[142] Lingzhu Xiang, Zhile Ren, Mengrui Ni, and Odest Chadwicke Jenkins. Robust graph slam in dynamic environments with moving landmarks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2543–2549. IEEE, 2015.

[143] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3D voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1903–1911, 2015.

[144] Jianxiong Xiao, Bryan C Russell, and Antonio Torralba. Localizing 3d cuboids in single-view images. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[145] Jia Xu, René Ranftl, and Vladlen Koltun. Accurate optical flow via direct cost volume processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[146] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 756–771. Springer, 2014.

[147] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 702–709. IEEE, 2012.

[148] Chun-Nam John Yu and Thorsten Joachims. Learning structural svms with latent variables. In *International Conference on Machine Learning (ICML)*, pages 1169–1176. ACM, 2009.

[149] AL Yuille and Anand Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

[150] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1994.

[151] Jian Zhang, Chen Kan, Alexander G Schwing, and Raquel Urtasun. Estimating the 3D layout of indoor scenes and its clutter from depth sensors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1273–1280. IEEE, 2013.

[152] Yinda Zhang, Mingru Bai, Pushmeet Kohli, Shahram Izadi, and Jianxiong Xiao. Deepcontext: Context-encoding neural pathways for 3D holistic scene understanding. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[153] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[154] Ziyu Zhang, Sanja Fidler, and Raquel Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 669–677, 2016.

[155] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems (NIPS)*, pages 487–495, 2014.

[156] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[157] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.