2D+3D Indoor Scene Understanding from a Single Monocular Image

Wei Zhuo

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

June 2018

Draft Copy – 27 June 2018

Declaration

Except where otherwise indicated, this thesis is my own original work. This thesis has not be submitted by me in whole or part for a degree or diploma in any university or other tertiary education institution. The content of this thesis is mainly based on my publications listed below:

- **(Chapter 3)** Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. "Indoor Scene Structure Analysis for Single Image Depth Estimation." In CVPR, pp. 614-622. 2015.
- **(Chapter 4)** Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. "3D Box Proposals from a Single Monocular Image of an Indoor Scene." In AAAI, 2018.
- **(Chapter 5)** Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. "Indoor Scene Parsing with Instance Segmentation, Semantic Labeling and Support Relationship Inference." In CVPR, 2017.

Wei Zhuo
27 June 2018

to my family.

Acknowledgments

I would like to express my greatest appreciation to all my supervisors for their great efforts on my PhD projects. They spent plenty of time in discussing with me about the research topics and the corresponding models. Without their devotion in time, insight and advices, it is not possible for me to complete my PhD projects.

I would like to thank my primary supervisor, Mathieu Salzmann, for his close supervision. Even after Mathieu moved to Switzerland, he keeps discussing with me regularly once a week. I would like to thank him for giving me great respect and freedom to select topics that I am interested in. He guided me to the field of scene understanding and taught me how to analyze and find approaches for the tasks. He is open-minded, active to give insights and keen to find problems. My PhD study benefits so much from him. I would like to thank Mathieu for his great patience as well, especially during my hard time on the second project. Furthermore, I would like to thank Mathieu for inviting me to visit EPFL, where I spent a good time with a group of excellent researchers in this world-leading institute.

I am also grateful to my supervisor, Xuming He. He taught me a lot during my PhD study. I would like to thank him for his valuable advices and insights, especially for my second and third publications. Without the plenty of discussion, it was not possible to get them published. I am also very grateful that he can spend a lot of time to discuss with me after he moved to the Shanghai Tech University of China, even though at that time he bore very heavy workload.

I would like to thank the third supervisor, Miaomiao Liu, for her great efforts and valuable time. I am grateful to Miaomiao that she devoted a lot of time to discuss with me about many details of everyday experiments, especially when Mathieu and Xuming are both on remote supervision. It is very helpful for me to keep a regular progress on my projects.

I would like to thank Hongdong Li, my panel chair. He helped deal with my academic schedule and always trust me and support me with a nice research environment. I appreciate his help and trust very much.

This research was supported by the scholarship provided by the Chinese Scholarship Council (CSC) and CSIRO. CSC provided the main part of my costs in Australia. CSIRO provided the supplementary scholarship, comfortable office environment, and experiemental equipments. I also would like to thank NVIDIA for the donation of a GPU processor.

Abstract

Scene understanding, as a broad field encompassing many subtopics, has gained great interest in recent years. Among these subtopics, indoor scene understanding, having its own specific attributes and challenges compared to outdoor scene understanding, has drawn a lot of attention. It has potential applications in a wide variety of domains, such as robotic navigation, object grasping for personal robotics, augmented reality, etc. To our knowledge, existing research for indoor scenes typically makes use of depth sensors, such as Kinect, that is however not always available.

In this thesis, we focused on addressing the indoor scene understanding tasks in a general case, where only a monocular color image of the scene is available. Specifically, we first studied the problem of estimating a detailed depth map from a monocular image. Then, benefiting from deep-learning-based depth estimation, we tackled the higher-level tasks of 3D box proposal generation, and scene parsing with instance segmentation, semantic labeling and support relationship inference from a monocular image. Our research on indoor scene understanding provides a comprehensive scene interpretation at various perspectives and scales.

For monocular image depth estimation, previous approaches are limited in that they only reason about depth locally on a single scale, and do not utilize the important information of geometric scene structures. Here, we developed a novel graphical model, which reasons about detailed depth while leveraging geometric scene structures at multiple scales.

For 3D box proposals, to our best knowledge, our approach constitutes the first attempt to reason about class-independent 3D box proposals from a single monocular image. To this end, we developed a novel integrated, differentiable framework that estimates depth, extracts a volumetric scene representation and generates 3D proposals. At the core of this framework lies a novel residual, differentiable truncated signed distance function module, which is able to handle the relatively low accuracy of the predicted depth map.

For scene parsing, we tackled its three subtasks of instance segmentation, semantic labeling, and the support relationship inference on instances. Existing work typically reasons about these individual subtasks independently. Here, we leverage the fact that they bear strong connections, which can facilitate addressing these subtasks if modeled properly. To this end, we developed an integrated graphical model that reasons about the mutual relationships of the above subtasks.

In summary, in this thesis, we introduced novel and effective methodologies for each of three indoor scene understanding tasks, i.e., depth estimation, 3D box proposal generation, and scene parsing, and exploited the dependencies on depth estimates of the latter two tasks. Evaluation on several benchmark datasets demonstrated the effectiveness of our algorithms and the benefits of utilizing depth esti-

mates for higher-level tasks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Everyday, as human beings, we move across various indoor scenes, e.g., living room, office, library and conference room, etc. We can easily perceive things in the room, reason about their relationships with ourselves and other surrounding objects, and interact with them. However, this remains a challenging task for a computer or a robot. Therefore, indoor scene understanding has gained great interest in decades. The ultimate goal of scene understanding is to facilitate a computer/robot to recognize categories of instances or objects, perceive their boundaries, and understand how these instances or objects are placed relative to each other. This information is extremely useful for grasping objects and navigation of personal robots, augmented reality, and, for outdoor scenes, automatic driving.

Indoor scene understanding, as a subcategory in the broad scene understanding field, has its own specific attributes and challenges. In contrast to outdoor scene images, indoor scene images mostly abide by the Manhattan world assumption [Coughlan and Yuille, 1999], such as the example in Figure 1.1, which assumes that the room is aligned with the three dominant and orthogonal directions, defined by the vanishing points. Based on this observation, previous work attempted to simplify the representation of indoor scenes by surface planes or 3D object boxes aligned to the three major directions [Hoiem et al., 2007a; Lee et al., 2009; Silberman et al., 2012a; Guo and Hoiem, 2013; Zhuo et al., 2015; Song and Xiao, 2016], or in some works with more strict constraints by a single box to represent a room [Hedau et al., 2009; Gupta et al., 2010a; Hedau et al., 2010; Schwing et al., 2013; Zhao and Zhu, 2013; Mallya and Lazebnik, 2015; Dasgupta et al., 2016; Ren and Sudderth, 2016]. Even though room scenes in reality usually bear more complex structures, the above approaches achieved promising performance by efficiently simplifying the problem. Beyond Manhattan world assumption, some works explored more flexible representations [Bansal et al., 2016; Eigen et al., 2014], with room layout embedded intrinsically. Despite the above attributes, the task of indoor scene understanding still suffers from great challenges: first, indoor scenes are usually cluttered and bear occlusions which makes it extremely difficult to infer the spatial layout and amodal shapes of objects; second, objects and instances of indoor scene cover a huge number of categories, and come in irregular shapes and sometimes very small sizes, which

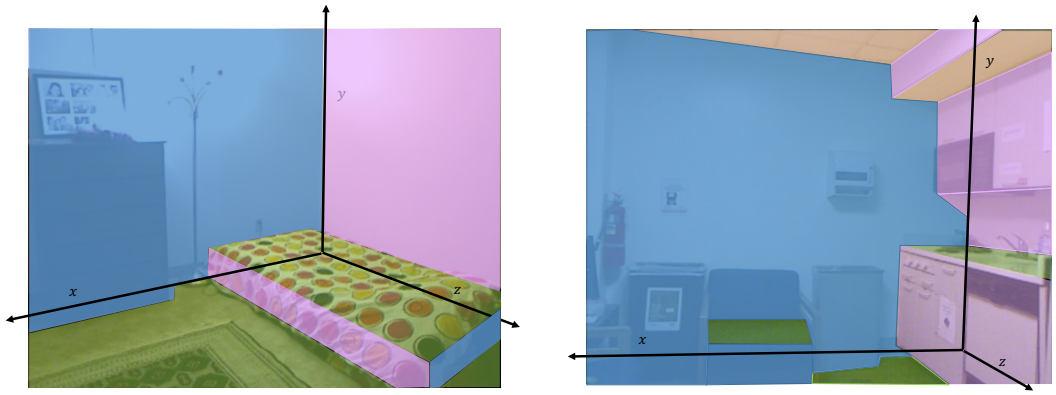


Figure 1.1: **Indoor scene image:** this example demonstrates the specific attributes of indoor scenes: a) The indoor scene is usually aligned with the three dominant and orthogonal directions; b) Most of its surface planes follow the three major directions of the scene.

(a) (b) (c) (d)

Figure 1.2: **Difficult samples of indoor scene:** these scenes are difficult to parse due to heavy occlusion (e,g. all the above examples), diverse object categories (e.g., example (b)) and indistinguishably appearance on permanent structures (e.g., the left and central wall of example (d).).

adds complexity to recognizing them; third, due to the imperfect illumination, indoor scenes usually bear indistinguishable geometrical boundaries along permanent structures, e.g., vertical walls of different orientations, especially when they have similar local appearances. For example, it is challenging to distinguish wall and ceiling when they have homogeneous color, which frequently happens for indoor scenes. Some difficult examples are shown in Figure 1.2. Because of the above mentioned intrinsic features and challenges of indoor scenes understanding, much research has been focusing on this task. In recent years, indoor scene understanding has achieved great success [Silberman et al., 2012a; Lin et al., 2013; Gupta et al., 2014a; Long et al., 2015; Gupta et al., 2015; Song et al., 2015; Song and Xiao, 2016]. However, to our best knowledge, most of these works made use of information from depth sensors, e.g., Kinect or LiDAR, to facilitate their specific task. Unfortunately, such sensors are not always available, or cheap enough to acquire. In this thesis, we dedicate our efforts to making indoor scene understanding work in the general case where only monocular color images are available.

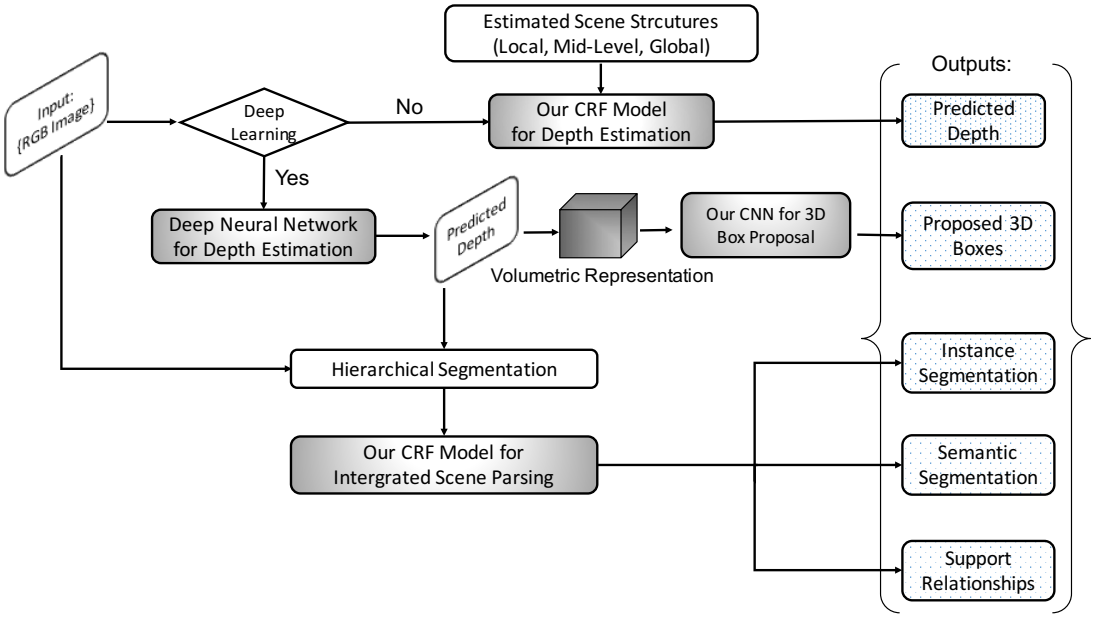


Figure 1.3: **Diagram overview of our methodology in this thesis.** From a single monocular color image, we tackled the following scene understanding tasks: a) We estimate a detailed depth map by leveraging scene structures on multiple scales independent of deep learning; b) Based on depth estimation using a deep convolution neural network (DCNN), we generate 3D box proposals by our residual, differentiable convolution neural network (CNN) acting on a volumetric representation of the scene; c) Based on depth estimation, we jointly reason about instance segmentation, semantic labels, and the support relationship of pair of segments in a hierarchical segmentation by our integrated CRF model.

1.2 Contributions

In this thesis, we tackle the problem of extracting 2D and 3D cues of indoor scene from a single monocular image. In particular, we focus on three fundamental tasks of indoor scene understanding from monocular images: depth estimation; 3D object box proposal generation; and scene parsing with semantic labels, instances and their support relationships. We diagram the framework of methodology in this thesis in Figure 1.3. First, to respect the fact that depth information provides complementary information to color images for high level recognition tasks, we devoted efforts into estimating depth maps from monocular images and introduced a depth estimation model which leverages high-level scene structures with local ones [Zhuo et al., 2015]. Second, based on predicted depth, we are able to generate 3D box proposals from a monocular image. In particular, we proposed a differentiable framework that predicts depth, generates a volumetric representation of a scene, and estimates 3D box proposals. Third, to provide a full understanding of the scene, we tackled the task of parsing a scene by segmenting it into meaningful regions, such as instances,

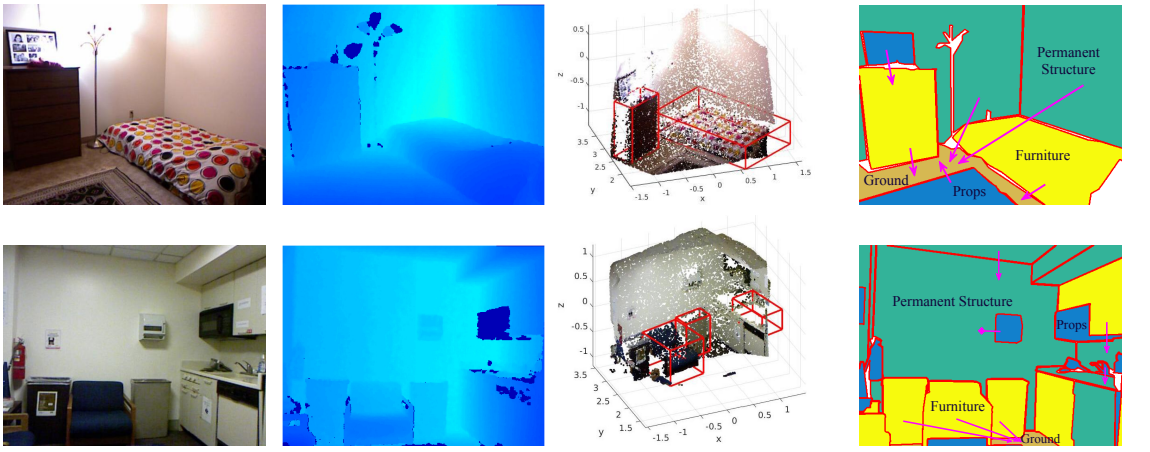


Image Depth 3D Object Boxes Semantic,Instance&Support

Figure 1.4: **Examples of our tasks for indoor scene understanding:** Here, we show the ground truth of a) depth (red is far, blue is close); b) some selected 3D object boxes in red; and c) scene parsing subtasks consisting of predicting semantics, instances and some selected support types on instances. In the right-most row, we outline instances by red line, label the four semantic categories with four different colours, and indicate that an instance is supported by another from below with a line with an arrow, and supported from behind by a line with a diamond.

predicting pixel-wise semantic labels and reasoning about the support relationships between arbitrary pairs of regions. To achieve this, we took advantage of the predicted depth to facilitate instance segmentation. More importantly, we exploited the mutual connections between the three subtasks: instance segmentation should respect semantic labels; support relations should depend on semantics and respect good instance regions [Zhuo et al., 2017]. An example is shown in Figure 1.4.

1.2.1 Depth Estimation with Scene Structure Analysis

Without any prior information, estimating the depth of a scene from a single monocular image is a highly ambiguous problem. Humans, however, can easily perceive depth from a static monocular input, thanks to the knowledge they accumulated over the years. Intuitively, this suggests that learning from existing image-depth pairs should make single image depth estimation a realistic, achievable goal.

Our model is motivated by the previous approaches to monocular depth estimation [Saxena et al., 2007, 2009; Liu et al., 2010; Karsch et al., 2012; Liu et al., 2014; Ladicky et al., 2014]. Before our work, these approaches typically reason about depth locally. However, we believe humans can perceive depth reliably with one eye thanks to high-level knowledge about scene structures, not local-scale information. In our work, we propose to exploit high-level scene structures for detailed depth estimation. To this end, we introduce an approach that relies on a hierarchical representation of the scene that models local, mid-level and global scene structures. More specifically,

our hierarchical representation includes different levels of information: superpixels, regions, and room layout. We formulate the single image depth estimation task as inferencing in a graphical model whose edges encode the interactions within and across different layers of the hierarchical representation. We experimentally demonstrate the benefits of exploiting high-level scene structure over local depth estimation methods.

1.2.2 3D Box Proposal from Monocular Image

Leveraging deep learning depth estimation approaches, e.g., [Eigen et al., 2014; Eigen and Fergus, 2015], we propose to predict 3D box proposals for objects based on single monocular image. Previous object proposal approaches based on monocular images typically reason about objects in 2D bounding boxes [Ren et al., 2015; Girshick, 2015; Dai et al., 2016; He et al., 2017]. By contrast, we argue that it is more natural to represent objects in 3D space. To this end, we developed an integrated, fully differentiable framework to reason about class-independent 3D box proposals from a single monocular image. In particular, we designed a deep network which consists of three modules, that predict a depth map [Eigen and Fergus, 2015], extract a volumetric representation of the scene, and generate 3D object proposals in the form of cuboids, respectively. At the core of our approach lies a novel residual, differentiable truncated signed distance function module, which accounts for the relatively low accuracy of the predicted depth map and extracts a 3D volumetric representation of the scene. To guarantee the effectiveness of our volumetric representation, we work on a multi-task learning scenario, where we estimate 3D boxes as well as the depth map simultaneously. To the best of our knowledge, this constitutes the first attempt to work in this challenging setting for complex indoor scene understanding. We experimentally demonstrate the effectiveness of our approach on 3D box proposal generation, as well as evidence the benefits of the different components of our framework.

1.2.3 Scene Parsing with Multi-Task Learning

To provide a full understanding of indoor scenes, we further focused on the task of indoor scene parsing. Indoor scene parsing is a complex problem that consists of multiple subtasks, such as segmenting a scene into regions that match instances [Arbeláez et al., 2014; Gupta et al., 2013; Shi and Malik, 2000], predicting pixel-wise semantic labels in the scene [Long et al., 2015; Eigen and Fergus, 2015; Zheng et al., 2015] and reasoning about the support relationships between arbitrary pairs of regions [Jia et al., 2013; Guo and Hoiem, 2013; Silberman et al., 2012a; Yang et al., 2017]. Specifically, the support relationships between regions indicate whether one region is supported by the other from below, or behind, or otherwise no support relationship [Silberman et al., 2012a; Zhuo et al., 2017]. These subtasks provide detailed category and boundary information about the instances, as well as how they interact with each other, and can thus have a great impact in a wide range of applications,

such as personal robotics, indoor navigation, indoor design, and augmented reality.

In the works mentioned above, these subtasks are usually tackled independently, despite the fact that strong connections exist among them. In addition, to our best knowledge, these works rely on depth information. To fully leverage connections between the subtasks, we introduced an integrated model that jointly reasons about instances, semantics and their support relationships, which work in this general scenario with only a single monocular image. In particular, by exploiting a hierarchical segmentation, we formulate our problem as that of jointly finding the regions that correspond to instances, and estimating their semantics and pairwise support relationships. We modeled this with a Markov Random Field, which allows us to further encode links between the different types of variables. The parameters were learned by a structural SVM framework. To complement the information of the monocular color image, we exploited the predicted depth [Eigen and Fergus, 2015] to facilitate our predictions, particularly that of instances and support relationships. The predicted depth facilitated instance segmentation by generating a better hierarchical segmentation respecting depth boundaries, in contrast to approaches based on only RGB images [Hoiem et al., 2011; Ren and Shakhnarovich, 2013]. The predicted depth also facilitated support inference, because the support of an instance strongly depends on the normal vector of its surface plane. We experimentally evidence that taking into account the dependencies between regions, their semantics and their support relations helps improving the prediction of the corresponding variables, with a particularly high impact on support relationships.

1.3 Thesis Outline

The remainer of this thesis is organized as follows,

- Chapter 2: Background. This chapter introduces the background theory of classification models we ultilized in this thesis. In particular, it includes the training and inference of graphical models, and the theory behind deep learning.
- Chapter 3: Depth Estimation from Monocular Images. This chapter introduces our novel depth estimation model with scene structure analysis.
- Chapter 4: 3D Box Proposals from Monocular Images. This chapter introduces our novel monocular image 3D box proposal generation to predict depth, generate a scene volumetric representation and propose 3D boxes for objects in a fully differentiable fashion.
- Chapter 5: Scene Parsing with Multi-Task Learning. This chapter introduces our novel integrated graphical model for instance segmentation, semantic prediction, and support relationship inference.
- Chapter 6: Conclusions and Future Work. In this chapter we conclude with a summary of our contributions, open issues, and future research directions.

Chapter 2

Background

This chapter introduces the background theory and models we employed in this thesis. In our work, we utilize graphical models and deep learning techniques to develop our models. In particular, considering the ability of graphical models to encode complex dependencies among different variables, we use conditional random fields (CRF) to encode the dependencies of variables in the tasks of depth estimation and scene parsing. For monocular image 3D object box proposal, we take advantage of deep learning techniques and build an integral deep convolutional neural network, which is trainable in an end-to-end fashion for multiple sequential tasks. In short, both CRF and deep learning play important roles in our work. Below, we review the fundamentals of CRF and deep learning to help the reader better understand the following chapters.

2.1 Conditional Random Field

CRF [Lafferty et al., 2001], a type of discrimitive probabilistic graphical model, is generally used for labeling, such as semantic segmentation [He et al., 2004; Silberman et al., 2012a], instance segmentation [Silberman et al., 2014], and discrete depth estimation [Karsch et al., 2012; Liu et al., 2014], to name a few. Instead of treating each variable in a graph independently, it respects the relationships among variables. By performing inference in the joint solution space over all variables, one can reach a high quality configuration for the graph as a whole.

A graph consists of nodes (also known as vertices) and edges which link nodes. In a probabilistic graphical model, each node represents a random variable, and the edges express dependencies among them. Let us denote observations by $X = \{x_1, x_2, ..., x_K\}$, and their corresponding variables by $\mathbf{Y} = \{y_1, y_2, ..., y_K\}$, where each variable y_i takes a value/label from the state set $\mathcal{L} = \{1, 2, .., N\}$. For example, in a segmentation scenario, the nodes can be superpixels in an image, observations X can be the appearance/features of superpixels such as color, and Y can be the semantic labels, e.g., floor, ceiling, walls. In this context, edges can be links between neighboring superpixels. Theoretically, a CRF captures the posterior joint probability distribution over all the random variables, given the observations, that is $P(\mathbf{Y}|X) = p(y_1 = l_1, y_2 = l_2, ..., y_K = l_k|x_1, x_2, ..., x_K), \forall l_1, l_2, ..., l_K \in \mathcal{L}$. By exploiting the notion

of Gibbs distribution [Koller and Friedman, 2009], this probability distribution can be defined by a corresponding energy function as

$$P(\mathbf{Y}|X) = \frac{1}{Z}\exp(-E(\mathbf{Y}, X)), \tag{2.1}$$

where E is an energy function for the graphical model, and Z, also known as the partition function, is a constant normalization. An energy function typically comprises unary terms, pairwise terms and high-order terms. Given observations for the nodes, 1) a unary term encodes the independent distribution of a variable over its possible states/labels; 2) a pairwise term encodes the relationships, or conditional dependencies, between two related variables ; 3) and a high-order term encodes complex relationships among the variables within a "clique" of size greater than 2. A "clique" is defined as a subset of nodes, where every two distinct nodes are connected by an edge. Let us denote $\mathcal{P}$ the set constructed by pairs of related nodes, and $\mathcal{C}$ the the set of cliques of size at least 3. In general, an energy function for a CRF can be formulated as

$$E(\mathbf{Y}, X) = \sum_{i}^{K} \phi_u(y_i, x_i) + \sum_{(i,j)\in\mathcal{P}} \phi_p(y_i, y_j, x_i, x_j) + \sum_{c\in\mathcal{C}} \phi_h(y_c, x_c) \tag{2.2}$$

where ϕ_u, ϕ_p, ϕ_h are called potentials. Given a CRF, or rather, a posterior joint probabilistic distribution on a graph, the ultimate goal is to find the set of state/label combination over all the variables, that achieves the highest joint probability, namely the lowest energy, over the graph, in the joint solution space. Let Y be a possible label configuration for all the variables. $\mathbf{Y}$, as defined above, is the set of Y, i.e., $\mathbf{Y}$ encompasses all possible configurations of all the variables. In practice, inference in a graphical model is formulated as

$$\hat{Y} = \arg\min_{Y\in\Omega} E(\mathbf{Y} = Y, X) \tag{2.3}$$

where Ω is the solution space, $\hat{Y}$ is the best configuration in solution space over all variables. The solution space of a CRF can have a number of N^K solutions at most, where, as mentioned before, N is the number of states for each variable, and K is the number of variables. When K and N are large, it is infeasible to solve the above optimization problem in a greedy manner. Therefore, many inference methods have been proposed for efficient and fast inference. It is worth noting that we assume each variable has the same number of states for simplicity. In practice, however, they can have different numbers of states. In the later chapters of this thesis, we assume all potentials depend on the observations, and omit the observations in our energy functions to shorten the notation.

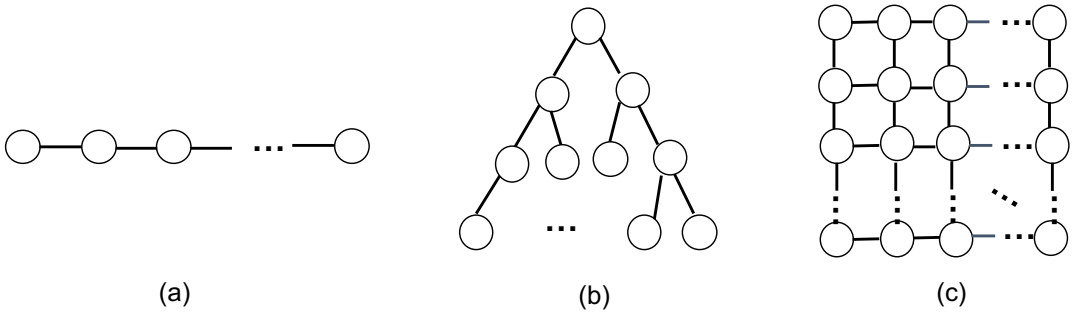


Figure 2.1: **Examples of graphs:** Here we show graphical structures in shapes of (a) chain, (b) tree, and (c) grid.

2.1.1 Inference

The goal of inference is to find the set of labels over all the variables, that minimizes the energy function of a graphical model. Many inference techniques have been developed to treat various graphical models [Boykov et al., 2001; Weiss et al., 2007; Krähenbühl and Koltun, 2011]. A complete review of inference techniques is beyond the scope of this chapter. We briefly introduce the ones we use in our work.

Exact inference is, in most cases, an NP-hard task. Inference on graphs with a chain or tree structure, however, can be solved exactly, the global optimum can be reached. Inference can be done by algorithms of searching shortest path, such as max-sum [Bishop, 2006], which is also known as Viterbi algorithm, an efficient dynamic programming algorithm. This procedure, however, is not natural to handle cases with non-local constrains on the outputs [Roth and Yih, 2005]. To handle the more general scenario that contains both local and non-local constraints, inference can be converted to an Interger Linear Programming (ILP) problem [Roth and Yih, 2005], where the variables are constrained to a limited number of discrete states, and abide some inequality, or equality constraints. In particular, it is in general represented as

$$\begin{aligned} \underset{\mathbf{Y}}{\text{minimize}} \quad & g_0(\mathbf{Y}) \\ \text{subject to} \quad & g_i(\mathbf{Y}) \leq 0, i = 1, 2, ..., N_c \\ & h_j(\mathbf{Y}) = 0, i = 1, 2, ..., M_c \\ & \forall y_i \in \{0, 1, ..., N\}, \\ & \text{where,} \mathbf{Y} = \{y_1, y_2, ..., y_K\} \end{aligned} \tag{2.4}$$

In our work [Zhuo et al., 2017], we exploited the special case where the variables are boolean. In fact, the general case of (2.4) can be converted to a boolean problem, by converting each y_i to an N-dimentional vector $a_i = [a_i^1, a_i^2, ..., a_i^N]$, such as $a_i^j \in \{0, 1\}$ and $\sum_{j=1}^{N} a_i^j = 1$. This ILP formulation provides a systematic tech-

nique for general scenarios, especially the ones with non-local constraints. In general, ILP problems are solved using a linear-programming based branch-and-bound algorithm [Nemhauser and Wolsey, 1988]. Existent commercial optimization toolboxes, such as Gurobi, can solve problems of this type in tractable time in many scenarios when the problem is properly formulated.

Inference in grid graphs, which contains cycles, is NP-hard and exact solutions cannot be obtained. Several inference techniques, however, exploit the graph structure to obtain approximate solutions, which are sovable in polynomial time. Belief Propagation (BP) is a classical iterative massage-passing technique for problems of this type. In BP, each node collects messages/marginal potentials from its neighboring nodes, except its direct descendant, and passes them to its descendants. With limited iterations, it can get an approximate lower bound of the energy function. In our work [Zhuo et al., 2015], we use the technique of Distributed Convex Belief Propagation (DCBP) [Schwing et al., 2011], which enables fast inference on a large graph due to its distributed parallel computation. In particular, it partitions a graph into several subgraphs and treats subgraphs, in parallel, as local optimization problems. It then imposes agreements on the beliefs at the junctions of these subgraphs.

2.1.2 Learning a Graphical Model - Structural SVM

To start with, we rewrite above energy function in Eq. (2.2) to a parametric form

$$\Psi(\mathbf{Y}, X) = -E(\mathbf{Y}, X) = \mathbf{w} \cdot \psi(\mathbf{Y}, X) \tag{2.5}$$

where $E(\mathbf{Y}, X)$ is the energy function on the variable set $\mathbf{Y}$, given the observations X, and $\Psi(\mathbf{Y}, X)$ is its corresponding negative, i.e., the score function. ψ is the vector concatenating all potential function values for a given $\mathbf{Y}$, and $\mathbf{w}$ is a parameter vector.

For a graph defined on an image i, state assignments for the graph variables can be divided into two categories, i.e., the ground truth assignment, Y_i, and non-ground-truth assignments, $Y \in \Omega_i \setminus Y_i$ (with Ω_i the assignment space for the graph defined on image i). We want to learn a parameter vector such that the ground-truth assignment has the lower energy, i.e., the higher score, than the non-ground-truth assignments.

$$\forall Y \in \Omega_i \setminus Y_i, \mathbf{w}^T(\psi(Y_i, X_i) - \psi(Y, X_i)) \geq 0 \tag{2.6}$$

To this end, we formulate this with probem by a hard-margin SVM. That is,

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ s.t. \quad & \forall i,\ \forall Y \in \Omega_i \setminus Y_i : \mathbf{w}^T(\psi(Y_i, X_i) - \psi(Y, X_i)) \geq 1 \end{aligned} \tag{2.7}$$

where the constraints are imposed on a set of image samples. To allow errors in training, the hard-margin optimization is relaxed with a soft-margin criterion and slack

variables. In the end, this yields an optimization problem, called n-slack structural SVM with margin-rescaling [Tsochantaridis et al., 2005] defined as

$$\begin{aligned} &\min_{\mathbf{w},\epsilon_1,\epsilon_2,\ldots,\epsilon_n} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\lambda}{n}\sum_{i=1}^{n}\epsilon_i \\ &s.t. \quad \forall i,\ \forall Y \in \Omega_i \setminus Y_i,\ \epsilon_i \geq 0 : \mathbf{w}^T(\psi(Y_i, X_i) - \psi(Y, X_i)) \geq \triangle(Y_i, Y) - \epsilon_i \end{aligned} \tag{2.8}$$

where $\triangle(Y_i, Y)$ is a task-dependent loss encoding the error of predicting output Y given the ground truth Y_i, which also works as a measure of the quality of a prediction. The loss follows the assumptions that: 1) ground truth should have zero loss, i.e., $\triangle(Y_i, Y_i) = 0$; 2) any non-ground-truth configuration has a positive loss, i.e., $\triangle(Y_i, Y) > 0, \forall Y \neq Y_i$. Training a structural SVM aims to learn the parameter vector that one pushes the score gap between an arbitrary predicted configuration and the ground truth to be larger than the loss. This is formulated as constraints in the problem of Eq. (2.8). This also allows the better predictions to have scores closer to that of the ground truth than the worse predictions. In this sense, structural SVM respects the ranks of the prediction quality.

Due to the combinatorial nature of the output space, there is an exponential number of constraints in Eq. (2.8), which makes optimization difficult. To reduce the constraints, at each iteration, the algorithm only considers a set of most "violating" output configurations, i.e., the ones that do not satisfy the constraints in Eq. (2.8). For the i-th graph, to find its most "violating" output configuration, we optimize a loss-augmented function

$$\widetilde{H_i}(\mathbf{w}) = \max_{Y \in \Omega_i} \triangle(Y_i, Y) - \mathbf{w}^T(\psi(Y_i, X_i) - \psi(Y, X_i)). \tag{2.9}$$

Given the violated outputs, problem (2.8) is solved using the primal-dual technique [Tsochantaridis et al., 2005]. A training procedure with faster speed was proposed by [Lacoste-Julien et al., 2013], which introduces a block-coordinate Frank-Wolfe algorithm to compute the optimal step-size and yields a duality gap gurantee. In practice, the parameters of (2.8) are trained in iterative manner on "hard" samples in a set of graphs, each of which is defined on an image. In particular, the "hard" samples, for which the current parameters work poorly, are selected based on $\widetilde{H_i}(\mathbf{w})$. Specifically, if $\widetilde{H_i}(\mathbf{w}) > 0$ the current parameters work poorly on this sample, and we need to use it to learn the parameters. Otherwise, all potential configurations satisfy the constraints, and learning can skip this sample.

2.2 Deep Learning

The research on convolutional neural networks (CNN) began decades ago [LeCun et al., 1990; Simard et al., 2003]. In recent years, CNNs have gone to extremely deep architectures, making breakthrough achievements on recognition tasks [Krizhevsky

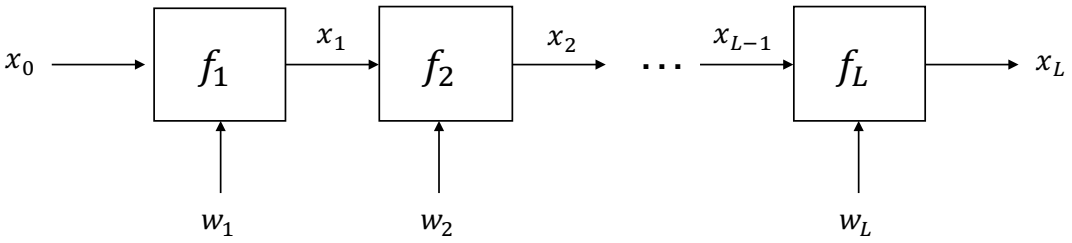


Figure 2.2: A neural network with chain-like architecture [Vedaldi and Lenc, 2015].

et al., 2012; Simonyan and Zisserman, 2014; Szegedy et al., 2015; He et al., 2016; Huang et al., 2017]. The success of CNNs can be attributed to three critical points. First, the successful deep convolutional neural networks (DCNNs) are carefully designed. DCNNs usually consist of stacking processing layers, typically convolutions and ReLUs, which enable them to generate features at multiple scales. The architecture ideas, such as inceptions and resnet, led to further significant development to the deep learning family. Second, a crucial data development of large supervised datasets makes efficient learning of deep network a reality. Early attempts to train deep nets with unsupervised data could hardly achieve performance comparable to current ones learned with supervised data. In the current era, however, large labeled datasets of tens of millions of images, e.g., ImageNet [Deng et al., 2009], makes it possible to learn a huge number of parameters. Third, the development of graphics processing units (GPU) enables intensive parallel computation in feasible time. Motivated by AlexNet [Krizhevsky et al., 2012], researchers exploited DCNNs to go deeper, wider, and work more efficiently. Deep learning techniques are still fast-developing, with novel neural units constantly added into the blocks of deep learning, and more powerful frameworks replacing existing ones. A full review of deep learning is beyond the scope of this section. We will focus on the fundamentals related to our work.

2.2.1 Fundamentals

Deep learning provides a powerful framework for supervised learning. In general, a convolutional neural network (convnet) is built as a cascade of basic computational blocks. Let us denote by $\mathbf{x}$ the input data and by $\mathbf{y}$ the predictions. In a practical segmentation scenario, for example, the input here can be an RGB image, and the predictions are a labeling map of the same resolution as the input [Long et al., 2015]. A neural network with a chain-like architecture, as shown in Figure 2.2, can be represented as a mapping from $\mathbf{x}$ to $\mathbf{y}$ [Vedaldi and Lenc, 2015],

$$\mathbf{y} = f(\mathbf{x}; \mathbf{w}_1, \cdots, \mathbf{w}_L) = f_L(\cdot\,; \mathbf{w}_L) \circ f_{L-1}(\cdot\,; \mathbf{w}_{L-1}) \circ \cdots \circ f_1(\mathbf{x}_0; \mathbf{w}_1), \tag{2.10}$$

where $\mathbf{x}_0 = \mathbf{x}$, $\mathbf{w}_l$ is parameters in the l-th layer, and $\circ$ is the composition operator. That is, for two functions, $f_l(\cdot\,; \mathbf{w}_l) \circ f_{l-1}(\mathbf{x}_{l-2}; \mathbf{w}_{l-1}) = f_l(f_{l-1}(\mathbf{x}_{l-2}; \mathbf{w}_{l-1}); \mathbf{w}_l)$. In

the mapping above, each layer acts on the output of its predecessor, $\mathbf{x}_l = f_l(\mathbf{x}_{l-1}; \mathbf{w}_l)$, depending on a set of parameters. Each layer, f_l, defines an operation, which can be anyone of convolution, pooling, rectified linear unit (ReLU) for activation function, batch normalization, fully connected layer, etc. Among these functions, convolutions depend on a filter size and a sliding stride and pooling also relies on a sliding stride. In general, the filters are defined in the 2D spatial domain, however, 3D extensions also exist for 3D data analysis. Each computational layer works on a local region of an input feature map for this layer, where this region is defined by the size and center of the local filter. When we trace back this region through the network to the input image, it would correspond to an area, which is denoted as "receptive field". The receptive field generally increments when a convnet gets deeper. This fact reflects the hierarchical nature of convnets, whose features at the bottom layers usually correspond to low level features such as edges or lines, while the top layers yield high level conceptual information.

In a convnet, the last layer is generally a mapping from a feature map to the target, e.g., a regression vector or classification scores. In supervised learning, the prediction layer is generally followed by a task-dependent loss function, $E(f(\mathbf{x}; \mathbf{w}_1, \ldots, \mathbf{w}_L), \mathbf{y})$, where $\mathbf{y}$ is the ground-truth target. The loss function can be, for example, L2 square loss for regression, or the cross-entropy loss for classification. Parameters of the convnet are learned by back propagating the prediction errors, defined by the loss function, to the input layer. For a convnet with a chain-like architecture, with no skip connection, the gradient of the loss function with respect to the parameters of an arbitrary layer is computed by applying the chain rule [Vedaldi and Lenc, 2015],

$$\begin{aligned}
&\frac{dE(f(\mathbf{x}_0; \mathbf{w}_1, \cdots, \mathbf{w}_L), \mathbf{y})}{d(\text{vec } \mathbf{w}_l)^T} \\
&= \frac{dE(\mathbf{x}_L = f(\mathbf{x}_0; \mathbf{w}_1, \cdots, \mathbf{w}_L), \mathbf{y})}{d(\text{vec } \mathbf{x}_L)^T} \times \\
&\times \frac{d \text{ vec } f_L(\mathbf{x}_{L-1}; \mathbf{w}_l)}{d(\text{vec } \mathbf{x}_{L-1})^T} \times \cdots \times \frac{d \text{ vec } f_{l+1}(\mathbf{x}_l; \mathbf{w}_{l+1})}{d(\text{vec } \mathbf{x}_l)^T} \times \frac{d \text{ vec } f_l(\mathbf{x}_{l-1}; \mathbf{w}_l)}{d(\text{vec } \mathbf{w}_l)^T},
\end{aligned} \tag{2.11}$$

where, vec$(\cdot)$ is the vectorizing operation. During training, the parameters can be updated according to the gradient by a variety of optimization techniques [Bottou, 2012; Nesterov, 1983; Qian, 1999; Zeiler, 2012; Duchi et al., 2011; Kingma and Ba, 2014]. Among them, the most commonly used one is stochastic gradient descent (SGD) defined as,

$$\begin{aligned}
d\hat{\mathbf{w}}_l &= \frac{1}{M} \sum_{i=1}^{M} \frac{dE(f(\mathbf{x}_0; \mathbf{w}_1, \cdots, \mathbf{w}_L), \mathbf{y})}{d(\mathbf{w}_l)^T} \\
\mathbf{w}_l^k &= \mathbf{w}_l^{k-1} - \epsilon_k \cdot d\hat{\mathbf{w}}_l
\end{aligned} \tag{2.12}$$

where $d\hat{\mathbf{w}}_l$ is the average gradient on M samples, $\mathbf{w}_l^k$ is the updated parameter vector

for the l-th layer at the k-th iteration. In general, with a proper learning rate ϵ, learning converges to a local optimum after many iterations, if SGD is applied.

In essense, supervised deep learning learns the data representation, and the classifier or regressor, in an integrated manner. Compared to traditional methods that learn independent classifiers on hand-crafted features, deep learning is able to learn superior classifier and representation that better match each other. Note that, here, we introduce the theory of convnet in chain architectures. In reality, however, different architectures can be exploited, whose forward and backward processes can be easily obtained from Eqs. (2.10) and (2.11).

2.2.1.1 2D

Here, we introduce deep learning blocks in 2D space. Let us denote an input feature map by $\mathbf{x} \in \mathcal{R}^{H \times W \times D}$, an output feature map by $\mathbf{y} \in \mathcal{R}^{H'' \times W'' \times D''}$, and a processing filter with kernel $\mathbf{f} \in \mathcal{R}^{H' \times W' \times D \times D''}$ and a bias vector $\mathbf{b} \in \mathcal{R}^{D'' \times 1}$. s_h and s_w are sliding strides on height and width, respecitively. We list some basic computational layers below.

- Convolution: $\mathbf{y}(i'', j'', d'') = \mathbf{b}(d'') + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D} \mathbf{f}(i', j', d') \mathbf{x}(s_h(i''-1) + i', s_w(j''-1) + j', d', d'')$
- ReLU: $\mathbf{y}(i'', j'', d) = \max\{0, \mathbf{x}(i'', j'', d)\}$
- Max Pooling: $\mathbf{y}(i'', j'', d) = \max_{1<i'<H', 1<j'<W'} \mathbf{x}(s_h(i''-1) + i', s_w(j''-1) + j', d)$

In addition, the fully connected layer can be though of as an extreme form of convolution, whose kernel covers the full feature map. A typical layer is generally constructed by several convolution filters, each followed by a ReLU and a max pooling followed sequentially. In deep learning, a convolution is, by default, followed by an activation function, such as ReLU, sigmoid, and tanh, to provide non-linearity. Convolutions in a layer generate feature maps of size similar to the input, while the max pooling downsamples the feature map for next layer to increase contextual information. In addition, there exist layers of other types, such as the inception module in GoogLeNet [Szegedy et al., 2015], and the residual layer in ResNet [He et al., 2016].

When there is enough computation power and a large set of data samples, we usually train the model with a minibatch which contains several samples. In this scenario, batch normalization [Ioffe and Szegedy, 2015] reduces the internal covariate shift, which facilitates fast and stable training. In some cases, it can replace dropout, which randomly drops a part of parameters during training to prevent overfitting. In particular, batch normalization normalizes each channel of the feature map, $\mathbf{x}$, by averaging over batch instances. Let M be the batch size, $\mathbf{x}$ be the input, and $\mathbf{y}$ be the

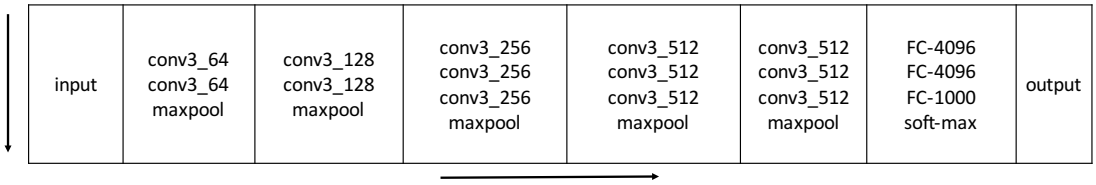


Figure 2.3: **VGG16:** The structure is shown from left to right, from top to bottom, where each convolutional filter is on size $k \times k \times c$, where k is the kernel size, c is number of channel. In this figure, the parameters of the convolution layer is shown in the format "conv<k>_<c>". Here, each convolution is be default followed by a ReLU activation function [Simonyan and Zisserman, 2014].

output, where $\mathbf{x}, \mathbf{y} \in \mathcal{R}^{H \times W \times D \times M}$. Batch normalization is formulated as

$$\begin{aligned} &\mathbf{y}(i,j,d,m) = w_d \frac{\mathbf{x}(i,j,d,m) - \mu_d}{\sqrt{\delta_d^2 + \epsilon}} + b_d, \\ &s.t. \quad \mu_d = \frac{1}{M}\sum_{m=1}^{M} \mathbf{x}(i,j,d,m), \quad \delta_d^2 = \frac{1}{M}\sum_{m=1}^{M} (\mathbf{x}(i,j,d,m) - \mu_d)^2 \end{aligned} \tag{2.13}$$

where w_d and b_d are the scale factor and a constant, for the d-th feature channel.

2.2.1.2 3D

In contrast to 2D convnets that work on images, 3D convnets work on volumetric inputs, such as a scene volume. Therefore, the filters in 3D convnets need one additional dimension. In general, they are straightforward extensions of the corresponding ones in 2D space. For example, a convolutional filter in 3D space is in five dimensions, that $\mathbf{f} \in \mathcal{R}^{H' \times W' \times L' \times D' \times D''}$, and performs the convolution in a similar manner to that in 2D space, but on a volume of four dimensions, to map a layer input of D' channels to an output of D'' channels. Due to the additional dimension, the memory consumption increases significantly to store the features and gradients. Therefore, with limited memory and time, it is hard to design a very deep 3D convnet. However, in contrast to 2D appearance features, a 3D convnet has the advantage of learning specific features in 3D, for example, 3D shapes in scene volumes [Song and Xiao, 2016].

2.2.2 Models

In this section, we introduce several networks, on which our models are based.

VGG [Simonyan and Zisserman, 2014]: [Simonyan and Zisserman, 2014] introduced several versions of very deep convolutional networks (VGG nets) of different lengths. In Figure 2.3, we show the structure and parameters of a VGG net with

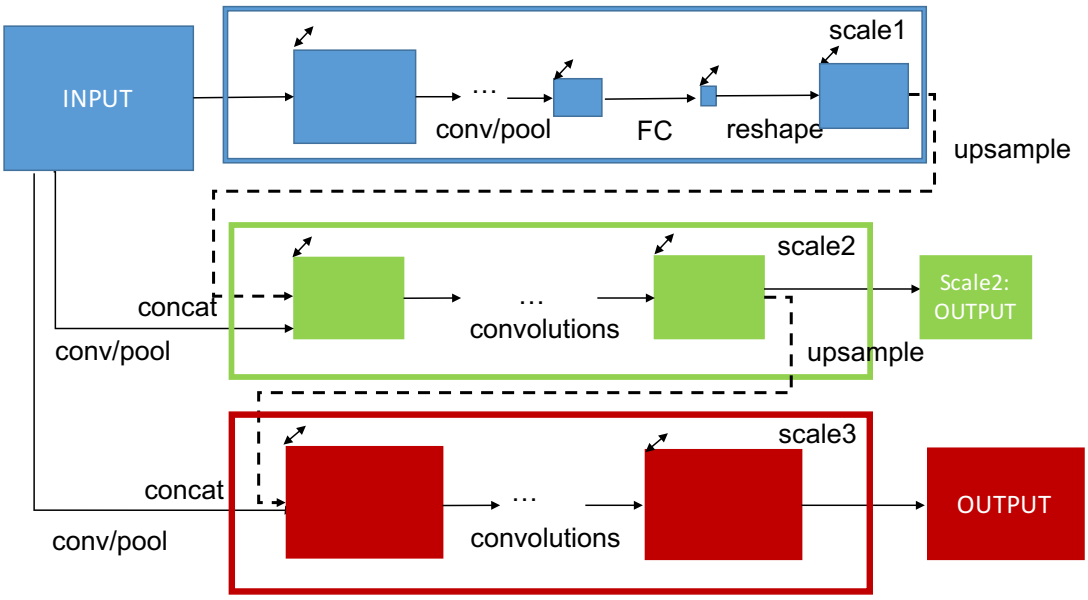


Figure 2.4: **Eigen's Network for Depth Estimation [Eigen and Fergus, 2015]:** This network is trained on images of size 240×320.

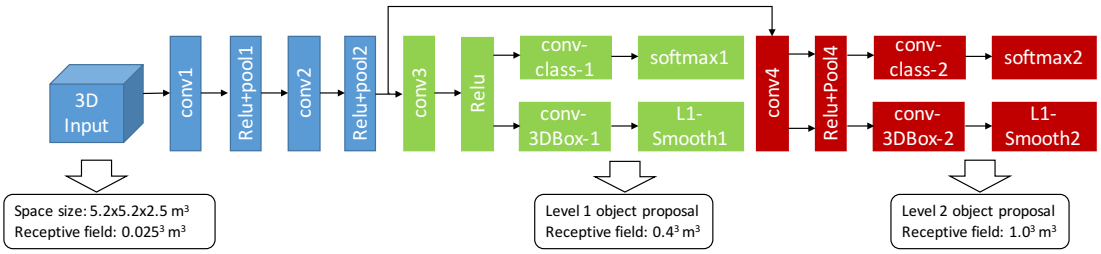


Figure 2.5: **Deep Sliding Shapes (DSS) [Song and Xiao, 2016]:** we show the structure and parameters of the DSS network in this figure. This network takes a 3D scene volume as input, and generates 3D object box proposals.

16 layers. In particular, the VGG16 consists of five convolutional blocks, followed by three fully connected (FC) layers. VGG nets were first introduced in [Simonyan and Zisserman, 2014] for classification on ImageNet, and the last fully connected layer outputs classification probabilities on 1000 categories. The classification loss is the cross-entropy loss. For other tasks, we can change the last layer to adapt to the specific number of outputs.

Fully Convolutional Neural Network (FCNN) [Long et al., 2015]: FCNN adapted the contemporary classification networks of [Krizhevsky et al., 2012; Simonyan and Zisserman, 2014; Szegedy et al., 2015] into fully convolutional networks for the segmentation task. In particular, for the model based on VGG16, it replaces the first fully connected layer (fc6) of VGG16 with a convolutional layer with kernel size

7×7. For image segmentation, the FCNN upsamples the predictions with a deconvolutional layer. For simplicity, the corresponding filter is set to be a bilinear filter. This network of linear topology is named FCN-32s. Based on this, [Long et al., 2015] developed directed acyclic graph (DAG) nets (FCN-16s and FCN-8s), with skip connections from lower layers to higher ones, aiming for predictions that leverage global information, as well as local details.

Eigen's Net [Eigen and Fergus, 2015]: [Eigen and Fergus, 2015] developed a multi-scale network based on the contemporary classification networks AlexNet and VGG for each of the tasks of image segmentation, depth estimation, and surface normal estimation from a monocular image. The architecture of this model is shown in Figure 2.4. In particular, this model consists of three scales: 1) the first scale generates a coarse predicted map at low resolution, but large receptive field. This part is typically an AlexNet or a VGG net, except for the FC layer; 2) scale 2 and 3, respectively, develop a shallow network on a larger feature map, with scale 3 contributing feature map of highest resolution. In contrast to FCNN, this network combines feature maps at multiple resolutions without sharing parameters.

Deep Sliding Shapes (DSS)[Song and Xiao, 2016]: DSS was developed to generate 3D box proposals, by taking a 3D scene volume as input, and predicting objectness and location parameters for each of the 3D boxes. In particular, a scene is divided into equally-spaced voxels with size of $X \times Y \times Z$. The model takes a scene volumetric representation in four dimensions, $\mathbf{x} \in \mathcal{R}^{X \times Y \times Z \times 6}$, as input, where each voxel in the 3D space has features of 6 dimensions, i.e., the signed truncated difference (TSDF) in x, y, z directions and the RGB values from the corresponding 2D location of the voxel. All its layers act in 3D. Each voxel of the output volume is associated with 19 anchors. These anchors represent potential 3D object bounding boxes of different sizes and aspect ratios. In practice, these anchors are divided into two categories by their size. For small anchors, the prediction is done from the bottom layers, which corresponding smaller receptive fields, while those of larger anchors are estimated by the top layers, which correspond to larger receptive fields. For each anchor, the network outputs a 2-dimensional probability vector for objectness, which is followed by a softmax loss, and 6-dimension estimations for the box center coordinates and size in 3D space, which is followed by an L1-smooth loss. The network architecture is shown in Figure 2.5.

2.3 Summary

In summary, in this chapter, we have introduced the fundamentals in CRF and deep learning that we exploited in our works in chapters 3-5. In particular, we presented the model representation, inference, learning procedure for CRFs, with an emphasis on the pipelines for inference and learning. For deep learning, we have presented the forward and backward process, and the architectures of several typical models.

Among the various types of computation layers in the deep learning family, we have selected the ones we utilised in our work. In short, this chapter has provided a brief background of the classical machine learning tools used in our work, and should help the reader gain a better understanding of the following chapters.

Chapter 3

Depth Estimation

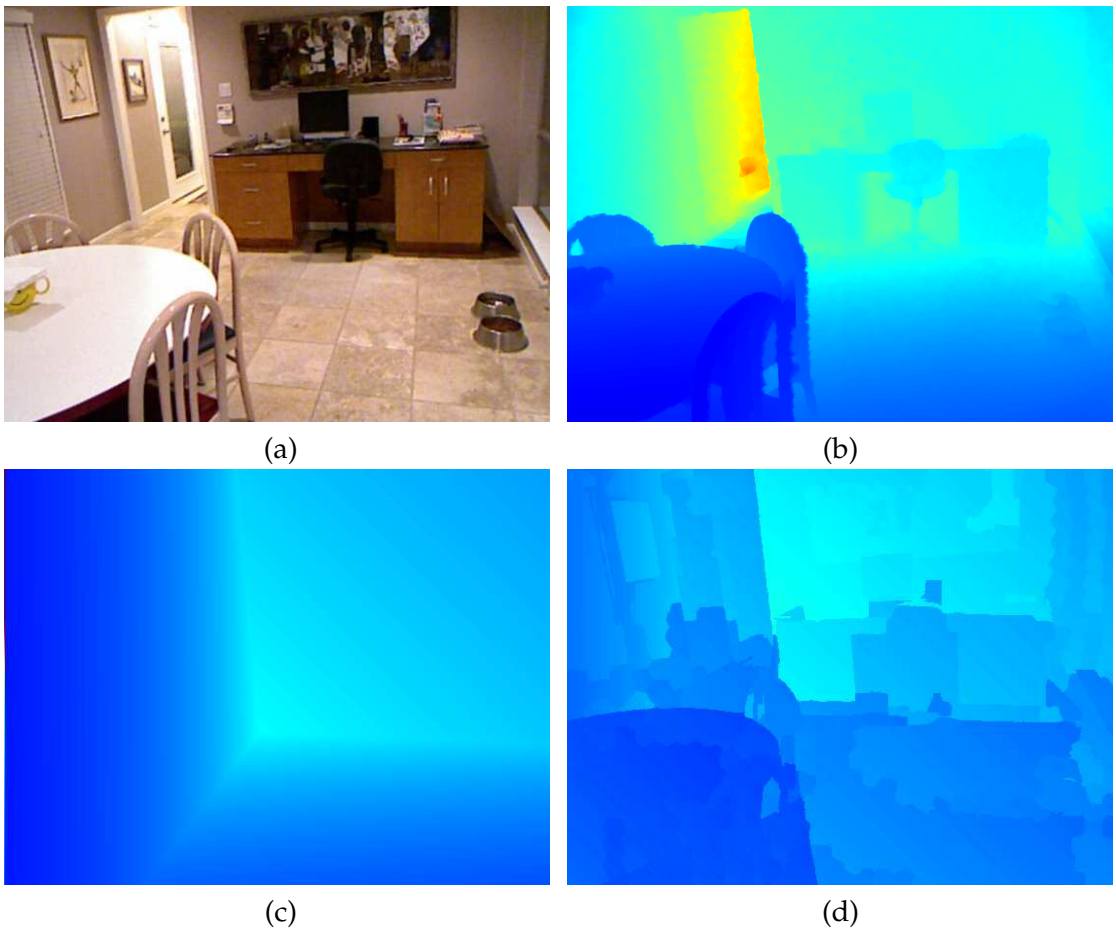
From this chapter on, we will introduce the technical work done during the course of this PhD. In particular, to understand a scene based on only a RGB image, we start with solving the problem of depth estimation from a single monocular image. As we know, depth information captures the distance of a 3D point to the camera, which, as a complement to the color image, provides critical information for robotics/computers to understand a scene in 3D. For example, depth is critical for perceiving the absolute range of a room and the locations of objects. With depth at hand, it is also easier to recognize occlusions, surface planes and corners, which usually pose great challenges to scene understanding based on only color images. Estimating depth facilitates overcoming the missing 3D geometric information, which can be applied in and benefitial for many applications, such as 3D movie production, augmented reality, robotic navigation, etc. In our work, i.e., Chapters 4 and 5, we also show that predicted depth can facilitate other scene understanding tasks, such as 3D box proposal generation and instance segmentation.

For the problem of monocular-image depth estimation, some approaches have achieved promising performance. Unlike previous approaches that only reason locally, we propose to exploit the global structure of the scene to estimate its depth. In particular, we propose a hierarchical representation of a scene at local, middle, and global level, and build a graphical model whose edges encode the interactions within and across the different layers of our hierarchy.

In the remainder of this chapter, we first introduce the motivation and idea in section 3.1, and the related work in Section 3.2. We then provide the details of our model in Section 3.3, and evaluate it with extensive experiments in Section 3.4. Section 3.6 concludes the chapter.

3.1 Introduction

Without any prior information, estimating the depth of a scene from a single image is a highly ambiguous problem. Early studies, however, have proved that humans have good abilities at perceiving ordinal depth and surface orientation from a single image [Cutting and Vishton, 1995; Koenderink et al., 1996; Koenderink, 1998; Zimmerman et al., 1995]. In addition to the much information provided by a monocular



(a) (b) (c) (d)

Figure 3.1: **Depth estimation from a single image:** (a) Image; (b) ground-truth depth map; (c) Estimated layout; and (d) detailed depth map. Color indicates depth (red is far, blue is close).

image, this ability is also due to their accumulative knowledge in real life. This suggests that learning from existing image-depth pairs could make single image depth estimation a realistic, achievable goal.

Motivated by the early studies, the earlier approaches [Saxena et al., 2007, 2009; Liu et al., 2010; Karsch et al., 2012; Liu et al., 2014; Ladicky et al., 2014; Eigen et al., 2014] have achieved promising performances by learning from a large amount of image-depth pairs. These methods, however, typically model depth only at a local scale. For instance, [Ladicky et al., 2014] predicts the depth of each pixel individually. While, in contrast, [Saxena et al., 2007, 2009; Liu et al., 2010; Karsch et al., 2012; Liu et al., 2014] encode some higher-level information by modeling the relationships of neighboring superpixels, the resulting methods still lack reasoning about the global structure of the scene. This contradicts our intuition that humans exploit such higher-level scene structure to apprehend their environment.

Recovering the structure of a scene has nevertheless been studied in the past [Ohta,

1985; Hoiem et al., 2005, 2007a; Lee et al., 2009; Gupta et al., 2010b,a; Hedau et al., 2010; Schwing et al., 2013; Fouhey et al., 2014]. The resulting methods typically represent the scene of interest at a coarse scale. As a consequence, they fail to provide a detailed description of the scene. More importantly, while these methods indeed infer the scene structure, they do not yield an absolute depth estimate; typically, only normals are predicted by these techniques, which leaves at least a global scale ambiguity for depth.

In this chapter, we propose to exploit high-level scene structure for detailed single image depth estimation. To this end, we introduce an approach that relies on a hierarchical representation of the scene depth encoding local, mid-level and global information. This lets us model the detailed depth of a scene while still benefiting from information about its global structure.

More specifically, our hierarchical representation of the scene depth consists of three layers: superpixels, regions and layout. The superpixels allow us to model the local depth variations in the scene. In contrast, the regions and layout let us account for mid- and large-scale scene structures. We model the depth estimation problem with a Conditional Markov Random Field (CRF) with variables for each layer in our hierarchy. This CRF allows us to encode interactions within and across layers, and thus to effectively exploit local and global information jointly. As illustrated in Fig. 3.1, inference in our model therefore yields depth estimates ranging from coarse to fine levels of details.

We demonstrate the effectiveness of our method on two standard indoor datasets. Our experiments evidence the benefits of exploiting higher-level scene structure over local depth estimation methods.

3.2 Related Work

In contrast to classical multiview approaches of 3D scene reconstruction, single image depth estimation has gained popularity only recently. While the 2D-to-3D mapping is an ill-posed problem, existing work have proven that it is an achievable goal. Among early works, some [Saxena et al., 2007; Karsch et al., 2012] assumed that images sharing similar scene appearance generally have similar depth maps at a coarse level, while others [Liu et al., 2010; Ladicky et al., 2014] exploited semantic constraints to improve depth. Recently, deep learning [Eigen et al., 2014; Li et al., 2015; Wang et al., 2015a; Liu et al., 2015; Roy and Todorovic, 2016; Laina et al., 2016; Xu et al., 2017] has led to great achievements in single image depth estimation, by intrinsically modeling the above mentioned assumptions. Among them, [Eigen et al., 2014] made a breakthrough improvement on depth estimation by applying deep neural nets to this task. Typically, the success of a deep learning model can be attributed to the hierarchical representation it encodes, which is usually implemented by stacking multiple convolutional layers that work on receptive fields of various sizes. However, before the emergence of deep learning, the majority of traditional methods reasoned about depth or scene structures at a single scale [Lee et al., 2009; Liu et al., 2014],

and the information of multi-scale space had not been well explored. In contrast to these methods, our depth estimation approach [Zhuo et al., 2015] models a scene representation at different scales, i.e., superpixels, regions and room layout, and reasons about scene structures as well as depth in one integrated model. In essense, our model follows an idea analogous to deep models that learn representations on receptive fields of various sizes, from small to large, and we demonstrate the effectiveness of this idea using hand-crafted features. Below, our literature review focuses on the methods based on traditional techniques, which fall in the same category as ours.

3.2.1 Depth Estimation

Due to the inherent ambiguities of the problem, existing approaches to monocular image-based depth estimation rely on training data in the form of image-depth pairs. In such a scenario, a natural approach is to learn regressors to predict local depth. This approach was employed in [Liu et al., 2010], where a specific regressor from image features to pixel-wise depth was trained for each semantic class in the dataset. This assumes that depth is constrained by semantic classes, e.g., sky is far away, and foreground objects are relative near, etc. Following a related idea, [Ladicky et al., 2014] trained classifiers for specific semantic labels at some chosen canonical depths. These classifiers were then employed to predict pixel depth. Several methods have proposed to go beyond purely local depth estimation. For instance, [Baig et al., 2014] introduced an approach based on sparse coding to directly predict the depth of the entire scene. Many techniques favor modeling the relationships between neighboring (super)pixels to encourage coherence across the image. With the exception of [Karsch et al., 2012; Konrad et al., 2012a,b] that formulate depth recovery as a purely continuous optimization problem, such coherence is typically encoded in a graphical model. This approach was introduced by [Saxena et al., 2007, 2009] with relatively simple relationships between the superpixels. A simple smoothness term was also employed in [Liu et al., 2010] together with local geometric reasoning and the previously mentioned regression as data term. In [Liu et al., 2014], additional discrete variables were employed to model more complex superpixel relationships, thus yielding a higher-order discrete-continuous graphical model. Despite the reasoning about neighbor interactions, all the above-mentioned models fail to consider the global structure of the scene, which provides important cues for depth estimation. To fully incorporate the structure context from local to global, our work [Zhuo et al., 2015] develops a CRF model reasoning about the depth of superpixels of roughly equal sizes, and of regions and room layout. To this end, our work takes advantage of the success of scene structure analysis to incorporate scene representations at multiple scales. In particular, we exploit mid-level regions and global scene layout to incorporate high-level representations of the scene.

3.2.2 Scene Structure Analysis

Before the era of deep learning, estimating the structure of a scene has itself been an active area of research, and can well assist the task of scene understanding. Here we

briefly review the work in this area and describe its relationship with our task. Dating from the 1960s, "blocks world" [Roberts, 1963] had been proposed to understand a scene. More recently, geometrical scene structures have been massively exploited. In this context, [Torralba and Oliva, 2002] modeled structure as the absolute mean depth of the scene. To model more detailed structure, many works originated from the idea of geometric context introduced in [Hoiem et al., 2005]. [Hoiem et al., 2005, 2007a] reasoned about scene geometric structures at multiple scales by grouping superpixels to three super classes and their vertical subclasses. It was later extended to predict the layout of indoor scenes with a box model [Hedau et al., 2010; Lee et al., 2010; Schwing et al., 2013], relying on the Manhattan world assumption [Coughlan and Yuille, 1999]. In particular, the box model represents the scene as five surface planes, i.e., ground, ceiling, and left, middle and right wall. To relax the strong constraint of the box model, forcing each pixel/superpixel belong to a surface, [Hedau et al., 2009] predicted the probability of a pixel/superpixel belonging to a foreground clutter. Instead of a box model, [Nedovic et al., 2010] classified a scene into 15 geometry categories to represent its structure. A more accurate representation was proposed in [Lee et al., 2009], and was able of producing surface normals on sparse segments. Similarly, in [Fouhey et al., 2013], local normals were predicted by exploiting Exemplar SVMs on detected discriminative regions. [Fouhey et al., 2014] improved such normal estimation by making use of a CRF and reasoning about normal discontinuities. [Ladickỳ et al., 2014] estimated dense surface normal by a local-coding regressor based on contextual and segment-based cues. These approaches usually grouped superpixels/regions according to their its surface normals. However, they lack information of absolute depths, which is critical for many application scenarios, such as 3D reconstruction. Instead, our work utilizes the scene structure to incorporate mid-level and global scene structures for detailed depth estimation. To this end, we introduce the hierarchical representation of the scene discussed in the next section.

3.3 Methodology

In this section, we introduce our hierarchical model to perform single image depth estimation. In particular, our model is built on three levels of scene structure representations, i.e., superpixels for the local level, regions for the middle level, and room layout for the global level. Our framework is illustrated in Figure 3.2.

As mentioned earlier, depth estimation is expressed as inference in a CRF. As discussed in Chapter 2, a graphical model generally consists of nodes and edges. The nodes are superpixels, regions, and the layout in our case. The edges, which connect the nodes, encode relationships within and between nodes of the different layers in our hierarchy framework. Inference in the graphical model allows the depth estimation to leverage detailed depth, as well as the mid-level and global scene structures. To this end, we denote by Y, R and L the variables that represent local depth, mid-level and global structures, respectively. Inference is achieved by maximizing

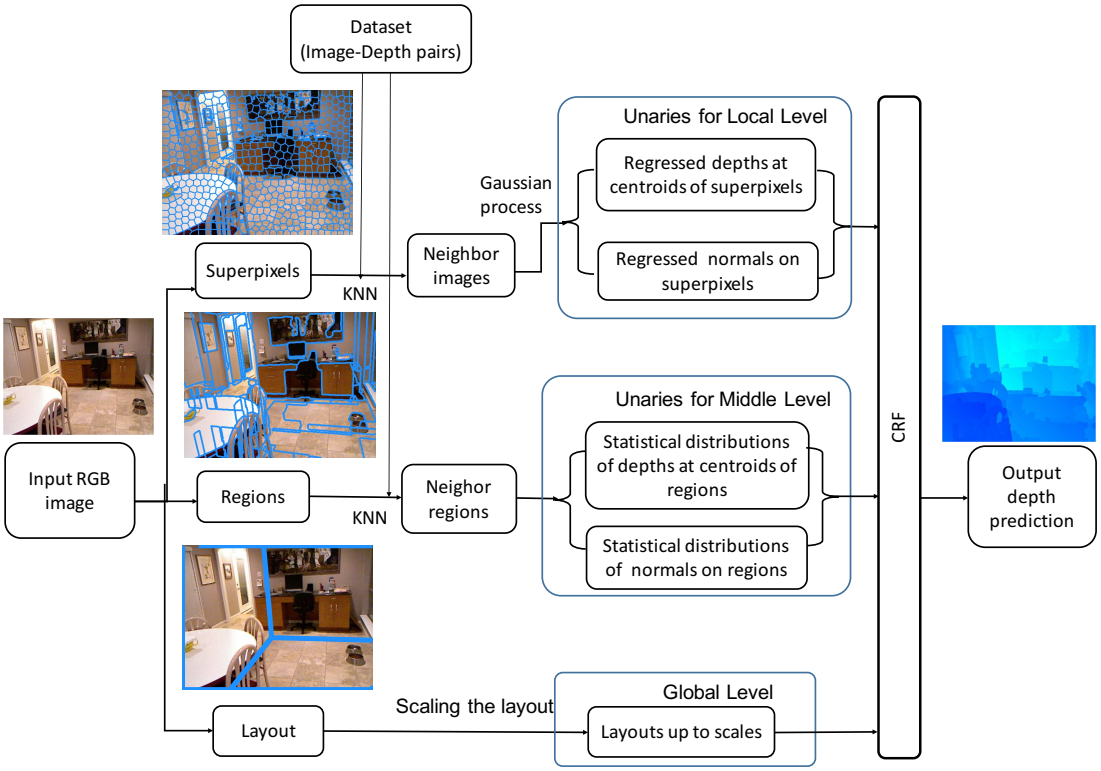


Figure 3.2: Our monocular depth estimation framework.

the joint distribution of our CRF, or equivalently minimizing the energy

$$E(Y, R, L) = E_l(Y) + E_m(Y, R) + E_g(Y, L) \,, \tag{3.1}$$

where each individual energy term corresponds to a particular layer in our model. In the remainder of this section, we describe these different terms in details.

3.3.1 Local Depth Estimation

To estimate detailed depth, our model relies on image superpixels. Each superpixel is represented as a plane in 3D, which translates the depth estimation problem into finding the best plane parameters for each superpixel. In particular, here, we encode each plane with the depth of its centroid and its normal direction.

More specifically, let $Y = \{y_1, y_2, ..., y_{N_s}\}$ be the set of discrete variables representing N_s superpixels in an image, where each y_i can take values from a discrete state space $\mathcal{S}$. We define this state space by quantizing the range of valid depth for the superpixel centroid into V values, with the range determined from the maximum and minimum depth of the training data. Furthermore, we make use of the Manhattan world assumption, and restrict the superpixels normal direction to 3 possible dominant directions, defined by the vanishing point estimation method of [Rother,

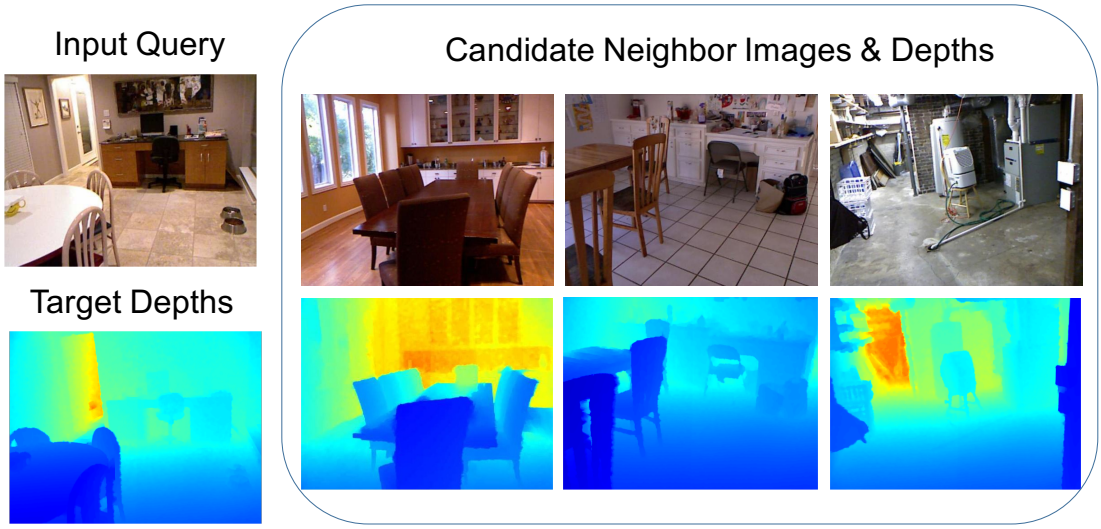


Figure 3.3: **An example for global neighbor search:** We show several neighbor images to a query image retrieved using image global context features. These neighbors depict either analogous layout, or objects similar to those in the input image.

2002]. This lets us define the first energy term in Eq. (3.1) as

$$E_l(Y) = \sum_p \phi_p(y_p) + \sum_{p,q} \phi_{p,q}(y_p, y_q) \,, \tag{3.2}$$

where ϕ_p is a unary potential encoding the cost of assigning label y_p to superpixel p, and $\phi_{p,q}(y_p, y_q)$ is a pairwise potential encouraging coherence across the superpixels.

The unary potential is based on the regression term in [Liu et al., 2014]. Due to the fact that superpixels occupy very small areas, and their features are insufficient to find the neighbors similar in depth, we rely on global context. To this end, we first retrieve K candidate training images similar to the input image by nearest-neighbor search based on a combination of distances on mutiple features, i.e., L_2 distances on GIST [Oliva and Torralba, 2001] and ObjectBank [Li et al., 2010] features, and χ^2-distance[1] on PHOG [Bosch et al., 2007]. Among these features, GIST and PHOG capture low-level scene representations, which summarize the gradient distribution of a scene image, while ObjectBank captures a high-level image representation, i.e., a response map to a set of pretrained object detectors. In practice, we select the candidate images from the training data in a leave-one-image-out manner. An example is shown in Figure 3.3. With the neighbor training images at hand, we compute, for each superpixel of the input image, the plane parameters of the corresponding area in each candidate, by fitting a plane from ground truth depths using RANSAC. We then use Gaussian Process (GP) regressors [Bishop, 2006] to predict the plane parameters of the superpixel of interest from these plane parameters (i.e., one regressor for each of the four plane parameters). The GP regressors rely on an RBF kernel.

[1]The χ^2-distance measures the difference between two vectors h and p as $\sum_i \frac{1}{2}(h_i - p_i)^2/(h_i + p_i)$.

(a) (b) (c)

Figure 3.4: **An example of boundary occlusion map:** (a) Image; (b) depth gradients, where color indicates gradient (red is large, blue is small).; and (c) ground truth boundary occlusion map. The boundary occlusion is labelled in purple.

Given the regressed depth on superpixel center and the regressed surface normal, we then can retrieve the depth of arbitrary points on the superpixel by the following procedure. Let $[n_x, n_y, n_z]$ be the normal vector, d_c the depth of the centroid of a superpixel. The corresponding 3D coordinates $[X_c, Y_c, Z_c]$ then are given by $d_c K^{-1}[x_c, y_c, 1]^T$, where $[x_c, y_c]$ are the 2D coordinates of the centroid, and K is the camera intrinsic matrix. Let $\mathbf{f} = [n_x, n_y, n_z, e]$ denote the paramters of a surface plane defined on the superpixel. As we know, for any point on a plane, we have $\mathbf{f}^T[X, Y, Z, 1] = 0$. Following this rule, we can calculate the fourth parameter of the plane as $e = -n_x x_c + n_y y_c + n_z z_c$. Then, for an arbitrary point on the superpixel, with 2D coordinates $[x, y]$, its depth d can be calculated as the intersection of the plane with its visual ray, i.e., $d = \frac{-e}{[n_x, n_y, n_z]^T K^{-1}[x, y, 1]}$.

Let $d^i_{r,p}$ be the depth of the i^{th} pixel of superpixel p, estimated from the regression results. We define our unary potential as

$$\phi_p(y_p) = \frac{1}{N_p} \sum_{i=1}^{N_p} \left(d^i_p(y_p) - d^i_{r,p} \right)^2 , \tag{3.3}$$

where N_p is the number of pixels in superpixel p and d^i_p is the depth of pixel i in superpixel p for a particular state y_p.

The pairwise term $\phi_{p,q}$ relies on an occlusion classifier trained on the features of [Hoiem et al., 2007b]. In practice, we assign non-occlusion/occlusion labels according to the average depth gradient on the edges, as shown in the example in Figure 3.4. The two-class classifier is trained based on boosted decision trees [Collins et al., 2002], with a non-occlusion/occlusion sampling ratio of 3.0. Given the predicted occlusion label o_{pq} for the boundary between two neighboring superpixels p

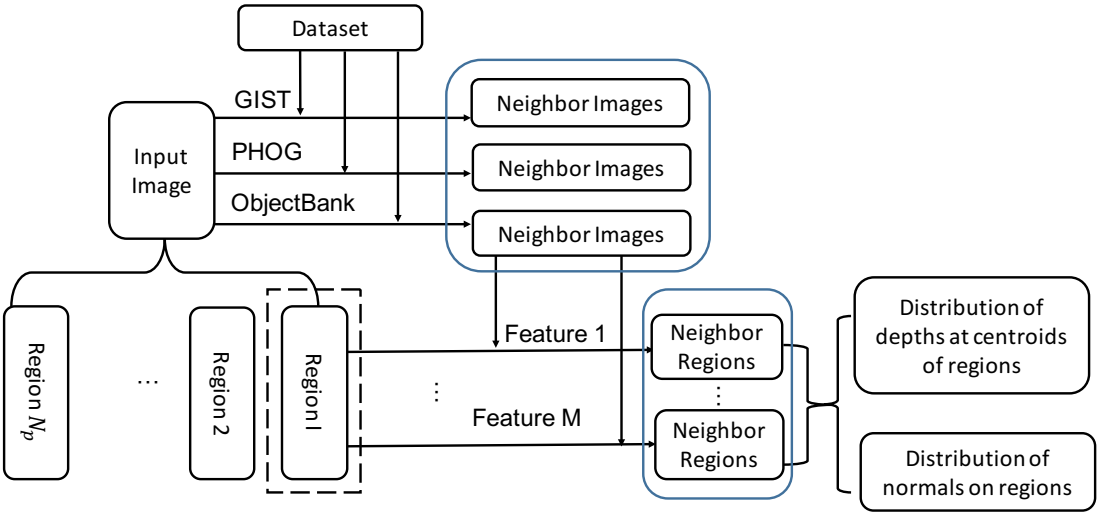


Figure 3.5: **Procedure for generating Mid-level unaries:** This depicts the precedure for generating mid-level unaries on regions.

and q, this potential is expressed as

$$\phi_{p,q}(y_p, y_q) = w_l \cdot \begin{cases} 0 & \text{if } o_{pq} = 1 \\ g_{pq}\|\mathbf{n}_p(y_p) - \mathbf{n}_q(y_q)\|^2 + & \\ \frac{1}{N_{pq}} \sum_{j=1}^{N_{pq}} (d_p^j(y_p) - d_q^j(y_q))^2 & \text{if } o_{pq} = 0 \end{cases} \tag{3.4}$$

where N_{pq} is the number of pixels shared by superpixels p and q, $\mathbf{n}_p(y_p)$ is the normal corresponding to a particular state y_p, and g_{pq} is a weight based on the image gradient on the boundary of the superpixels, i.e., $g_{pq} = \exp(-\mu_{pq}/\sigma)$ with μ_{pq} the mean gradient on the boundary. We assume that two neighbor superpixels have more chance to belong to two hinged planes, while the image gradient on the boundary is large, and in this case we penalize less their surface normal difference.

While inspired by [Liu et al., 2014], the energy described above includes at most pairwise terms, and therefore allows us to perform inference more efficiently. Importantly, however, this energy still reasons at a local level. In the following, we present our approach to incorporate higher-level scene structures via the additional terms in Eq. (3.1).

3.3.2 Exploiting Mid-level Structures

The superpixels employed above are typically quite small and therefore encode little information about the scene. As a consequence, not only do they encode little structure, but one also cannot reliably exploit their appearance to help depth prediction. Thus only their location in the candidate images retrieved using global image descriptors is utilized in the previous model. To better exploit appearance and encode more information about the scene structure, we propose to make use of larger

regions.

To this end, let $R = \{r_i, r_2, ..., r_{N_r}\}$ be the set of discrete variables representing N_r regions extracted from the input image, where each r_i can be assigned a value from the same state space $\mathcal{S}$ as the superpixel variables $\{y_p\}$. We define the second term in Eq. (3.1) as

$$E_m(Y, R) = \sum_{\gamma} \phi_\gamma(r_\gamma) + \sum_{\gamma,p} \phi_{\gamma,p}(r_\gamma, y_p) \,, \tag{3.5}$$

where ϕ_γ is a unary potential on the region variables, and $\phi_{\gamma,p}$ a pairwise potential accounting for the interactions of the regions and the superpixels.

Since our regions are much larger than our superpixels, their appearance is also more discriminative. Therefore, we follow a feature-based nonparametric approach inspired by [Tighe and Lazebnik, 2010] to define the unary term ϕ_γ. The procedure generating region unaries is illustrated in Figure 3.5. In particular, we first retrieve K_r candidate training images by nearest neighbor search using image-level GIST, PHOG and ObjectBank features. Here, we select the K_r images based on their best rank after nearest-neighbor search on each feature type individually. We found this strategy to be more reliable than combining the features for large retrieval sets. For each region in the input image, we compute region-level features[2] and retrieve K_c nearest-neighbor regions from the candidate image pools for each feature type, after pruning the ones that are too distant from and with too dissimilar sizes to the query region. Each superpixel in each retrieved region then votes for a centroid depth and a normal orientation in a V-dimensional and a 3-dimensional histogram, respectively. Let us denote by $P_d(d)$ and $P_n(\mathbf{n})$ the resulting normalized histograms and $P_{dn}(d, \mathbf{n})$ the bin-wise product of $P_d(d)$ and $P_n(\mathbf{n})$ (i.e., a 3V-dimensional histogram). We express the unary term in Eq. (3.5) as

$$\phi_\gamma(r_\gamma) = w_m \cdot \left(\max\left(P_{dn}(d(r_\gamma), \mathbf{n}(r_\gamma))\right) - P_{dn}(d(r_\gamma), \mathbf{n}(r_\gamma))\right),$$

where $d(r_\gamma)$ is the centroid depth corresponding to the state r_γ, and similarly for the normal direction.

The pairwise term in Eq. (3.5) penalizes inconsistencies between the depth predicted for a region and the depth predicted for the superpixels it covers. For each superpixel in a region, this term is defined as

$$\phi_{\gamma,p}(r_\gamma, y_p) = \frac{w_{m,l}}{N_p} \sum_{i=1}^{N_p} \left(d_p^i(y_p) - d_\gamma^i(r_\gamma)\right)^2, \tag{3.6}$$

where N_p is the number of pixels in superpixel p, and, with a slight abuse of notation w.r.t. index i, $d_p^i(y_p)$ and $d_\gamma^i(r_\gamma)$ represent the depth of the i^{th} pixel in superpixel p and of its corresponding pixel in region γ.

Note that the energy for the mid-level structures can be thought of as encoding longer range connections between the superpixels. Importantly, however, the resulting model remains pairwise.

[2]We used the same 20 features as in [Tighe and Lazebnik, 2010].

Extracting Regions

Here, we briefly describe our strategy to extract the regions acting as mid-level structures in the potentials described above. Our goal is to obtain regions that are preferably (close to) planar, of relatively uniform appearance and as large as possible. To this end, we rely on the gPb+Segmentation framework of [Arbelaez et al., 2011].

Since our training data consists of RGB-D images, we can directly employ the RGB-D extension of the gPb+Segmentation framework, introduced recently by [Gupta et al., 2014b; Ren et al., 2012]. At test time, however, we only have access to RGB images. An easy way around this problem would be to directly employ the original method of [Arbelaez et al., 2011]. Unfortunately, the resulting regions are either highly non-planar, or too small, both of which make them ill-suited for our purpose.

To address this issue, we propose to compute the probability of a boundary by combining two different sources of information. First, we rely on the standard gPb algorithm applied to our RGB input image. As a second source of information, we make use of the estimated scene geometry in the form of the orientation map of [Lee et al., 2009]. Orientation maps assign one major normal direction to the pixels in an image. Unfortunately, these maps are sparse (i.e., not all pixels are assigned an orientation). Furthermore, for our purpose, we would not want to have all pixels with the same orientation to belong to the same region, since they could potentially belong to different surfaces. Therefore, we compute the connected components of the orientation maps, and assign a label to each pixel indicating the component it belongs to. We then apply the gPb algorithm with brightness features only to the resulting label image.

Let us denote by gPb_{rgb} and gPb_{g} the boundary probabilities obtained from the RGB image and the geometry image, respectively. The combined boundary probability of a pixel at location (u, v) for boundary orientation θ is then given by

$$\mathrm{gPb}_{c}(u, v, \theta) = (1 - \alpha)\mathrm{gPb}_{rgb}(u, v, \theta) + \alpha\mathrm{gPb}_{g}(u, v, \theta),$$

where, in practice, we use $\alpha = 0.5$. To obtain the final regions, we then apply the OWT-UCM method of [Arbelaez et al., 2011] with a threshold of 0.1 on this combined boundary map. We found this combination of RGB and geometry cues to yield large, planar and uniform regions, well-suited for our approach. An example of our regions is shown in Figure 3.2.

3.3.3 Incorporating Global Structure

As a final layer in our representation, we aim to reason about the global structure of the scene, which neither the superpixels, nor the regions are able to model. To this end, we make use of the layout estimation method of [Hedau et al., 2009]. This method models the geometry of an indoor scene as a box made of five surfaces (i.e., left/middle/right wall, ceiling and floor), with an additional prediction of the probability of each pixel to belong to clutter. A layout example is illustrated in Figure 3.2, where the scene is modeled by three major surface planes, i.e., floor,

left wall, right wall. The normal vectors of the surface planes are estimated by the vanishing points. Note, however, that the output of this method is not truly a 3D representation, in the sense that the global scale of the box is not determined.

To make use of such global structure, let us denote by L the discrete variable encoding the scale of the predicted layout, which can take value in a state space $\mathcal{L}$ representing quantized scales. The energy for the last layer in our model can be written as

$$E_g(Y, L) = \sum_p \phi_{L,p}(L, y_p) \,, \tag{3.7}$$

and thus consists of a single pairwise potential that encourages coherence between the superpixels and the layout. In particular, we define this potential as

$$\phi_{L,p}(L, y_p) = \frac{w_g}{N_p} \sum_{i=1}^{N_p} (1 - P_c^i) \cdot (d_p^i(y_p) - d_L^i(L))^2 \,, \tag{3.8}$$

where P_c^i denotes the probability of pixel i belonging to clutter, and, with a similar slight abuse of notation w.r.t. index i as before, $d_p^i(y_p)$ and $d_L^i(L)$ represent the depth of the i^{th} pixel in superpixel p and of its corresponding pixel in the layout. The use of the clutter probability prevents us from oversmoothing the depth predicted by the superpixels.

Since this energy term is pairwise, so is our entire model. In our experiments, we make use of the Distributed Convex Belief Propagation (DCBP) method of [Schwing et al., 2011] to perform inference in our CRF. Note that the inference results yield not only a detailed depth estimate coming from the superpixels, but also an estimate of the region depths, as well as a full 3D layout of the scene.

3.4 Experiments

We evaluated our approach on two publicly available datasets: the NYUv2 depth dataset [Silberman et al., 2012b] and the RMRC Indoor dataset [RMR, 2014]. These two datasets both contain images collected from a wide variety of indoor scenes. For NYUv2, we compare our results with the state-of-the-art contemporary single image depth estimation methods. In particular, we consider the following three baselines:

1. **DepthTransfer [Karsch et al., 2012].** This method predicts depth by transferring depth maps from similar images in the training set. These depth maps are then merged by a continuous optimization strategy that encourages smoothness across the image.

2. **DC-Depth [Liu et al., 2014].** This technique makes use of a high-order discrete-continuous CRF to estimate depth, where complex relationships between the neighboring superpixels can be encoded via discrete variables.

3. **SemanticDepth [Ladicky et al., 2014].** This method learns a pixelwise classifier for each semantic class in the dataset at canonical depth, and therefore makes

use of an additional source of information in the form of semantic pixel labels. Besides, it was trained on a different training/test partition from the one provided with the dataset. Therefore comparison against their results is to take with a pinch of salt.

For the sake of completeness, we also report the results of the DeepDepth method of [Eigen et al., 2014]. This method, however, relies on a much larger training set consisting of the 120K raw images of the NYUv2 dataset and thus should not be considered as a true baseline.

In addition to the comparison with these methods, we also perform an ablation study where we provide the results of our local model (Section 3.3.1), the model that consists of local model and mid-level structures (Sections 3.3.1 and 3.3.2), and the model that consists of local model and global structure (Sections 3.3.1 and 3.3.3). We refer to these models as **Ours-local**, **Ours-mid** and **Ours-global-only**, respectively. Our complete model will be referred to as **Ours**.

For quantitative evaluation, we report the following three standard metrics: average relative error (**rel**), average $\mathbf{log}_{10}$ error, and root mean squared error (**rms**). In addition, we also report the metrics used in [Ladicky et al., 2014]. These metrics are defined as

1. average relative error (**rel**): $\frac{1}{N}\sum_u \frac{|g_u - d_u|}{g_u}$
2. average $\mathbf{log}_{10}$ error: $\frac{1}{N}\sum_u |log_{10}g_u - log_{10}d_u|$
3. root mean squared error (**rms**): $\sqrt{\frac{1}{N}\sum_u (g_u - d_u)^2}$
4. % correct: $\left(\frac{1}{N}\sum_{u=1}^{N}[[\max(\frac{d_u}{g_u^*}, \frac{g_u^*}{d_u}) = \delta < t]]\right) \cdot 100$, with $t = 1.25, 1.25^2, 1.25^3$

where g_u is the ground-truth depth at pixel u, d_u is the corresponding estimated depth, N is the total number of pixels in all the images, and $[[\cdot]]$ denotes the indicator function. Furthermore, even though normal estimation is not the main target of our approach, we report the five normal error metrics used in [Fouhey et al., 2013]: the **mean** and **median** angle difference between the estimated normals and the ground-truth ones, and the percentage of pixels whose angle difference w.r.t. ground-truth is below a threshold (i.e., $\theta < 11.25$, 22.5 and 30 degrees). To evaluate these metrics, the scene normals were estimated from the predicted depth maps by the method of [Fouhey et al., 2013].

In our experiments, the superpixels were computed using SLIC [Achanta et al., 2012]. In particular, we generate 638 superpixels per image on average. We find that this configuration yields an affordable computation cost and reasonable superpixel sizes to avoid boundary crossing. For each test image, we retrieve $K = 7$ candidates from the training images to obtain the input to the superpixel regression model. The occlusion classifier on boundaries of superpixels was trained on the standard

Method	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
DepthTransfer	0.374	0.134	1.12	49.81%	79.46%	93.75%
DC-Depth	0.335	0.127	1.06	51.55%	82.32%	95.00%
SemanticDepth	-	-	-	**54.22**%	82.90%	94.09%
Ours	**0.305**	**0.122**	**1.04**	52.50%	**83.77**%	**96.16**%

Table 3.1: **NYUv2: Comparison of our approach with the baselines on depth.** In terms of depth accuracy, we outperform the two baselines (DepthTransfer and DC-Depth) working under the same settings as us. Furthermore, we outperform the SemanticDepth approach on two out of three thresholds, despite the fact that we do not make use of any pixel label information. Recall, however, that SemanticDepth employed a different training/test partition.

training set of the NYUv2 dataset [Silberman et al., 2012a], and achieves an average accuracy of 72.1% on the NYUv2 test images. For the regions, we retrieve $K_r = 250$ candidate images. For each query region and each local features, we then obtain $K_c = 30$ candidate regions, after pruning the candidate regions whose centroid is at a distance $d > 100$ pixels from the query region centroid, and whose area ratio ($r_{area} = 2(area_a - area_b)/(area_a + area_b)$) with the query region is smaller than 0.2. When building the histogram of normal orientations, we only take into account the superpixels whose angle difference is less than 45 degrees w.r.t. at least one of the three dominant normal directions in the query image. This allows us to discard the candidates that have an orientation too different from the scene in the query image.

The states of our superpixel and region variables were obtained by quantizing the depth from 0.5 to 10 by steps of 0.5 (i.e., $V = 20$). In conjunction with the 3 normal orientations, this yields 60 states for each variable. In practice, to speed up inference, we restrict the states to the 20 values with highest probability P_{dn} in the $3V$-dimensional histogram built for the region unary potential. This speed-up brings very little loss of accuracy in the final results. In this setting, and given the result of gPb, estimating the depth of an image containing roughly 650 superpixels takes about 2 minutes.

The parameters of our CRF (weights of the potentials) were obtained by validation on a set of 69 images taken among the training data. To this end, we followed a strategy where the potentials were incrementally added to the energy after the previous weights were determined. Note that we did not fine-tune the weights, but mostly found the right order of magnitude of each potential among the values $\{0.1, 1, 10, 100, 1000\}$.

3.4.1 Evaluation on NYUv2

The NYUv2 depth dataset contains 1449 pairs of aligned RGB and depth images, partitioned into 795 training images and 654 test images. These images were acquired in a variety of real-world indoor scenes. Each image was cropped to 427×561 pixels. In our evaluation, we make use of a mask in each image that only considers the

Method	mean	median	$\theta < 11.25$	$\theta < 22.5$	$\theta < 30$
DepthTransfer	**43.0**	**40.5**	6.9%	23.2%	34.9%
DC-Depth	45.7	42.2	19.7%	25.7%	35.4%
Ours	46.7	41.9	**21.1%**	**35.2%**	**41.7%**

Table 3.2: **NYUv2: Comparison of our approach with the baselines on normal.** In terms of normal accuracy, we outperform the baselines on three out of the five metrics.

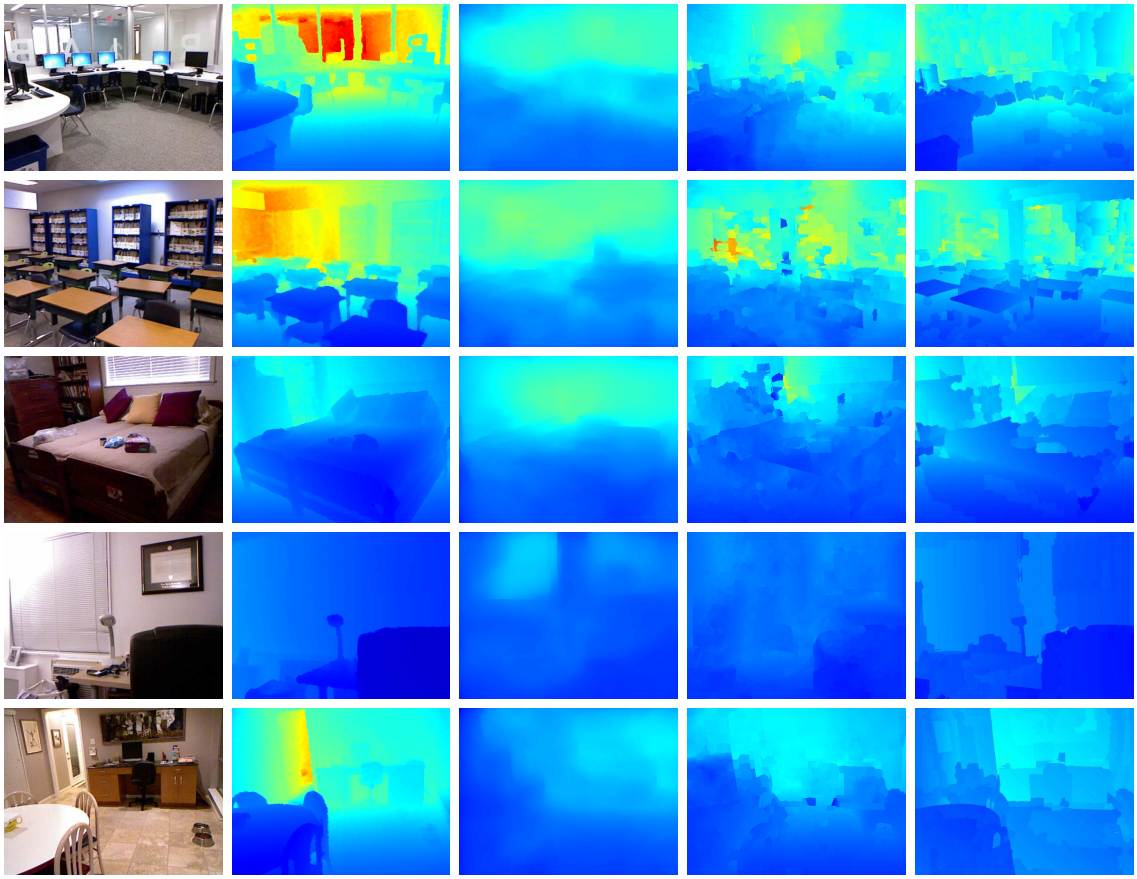


Figure 3.6: **NYUv2: Qualitative comparison.** Depth maps estimated by the different baselines and by our approach. Note that our approach typically avoids the oversmoothing of DepthTransfer, while better modeling the scene structure than DC-Depth.

ground-truth pixels with non-zero depth.

The results of our approach and of the baselines are shown in Table 3.1. In terms of depth accuracy, our approach outperforms DepthTransfer and DC-Depth on all error metrics, and SemanticDepth on two out of three threshold values, despite the fact that it exploits the additional knowledge of pixel labels during training. The

Method	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Ours-local	0.334	0.128	1.05	50.35%	82.31%	95.44%
Ours-mid	0.312	0.123	**1.03**	52.08%	**83.92**%	96.13%
Ours-global-only	0.325	0.128	1.07	50.38%	82.06%	95.35%
Ours	**0.305**	**0.122**	1.04	**52.50**%	83.77%	**96.16**%

Table 3.3: **NYU v2: Ablation study.** We evaluate the influence of the different components of our model. These results confirm that each parts of our model contributes to the final results, with a strong influence of the mid-level structures.

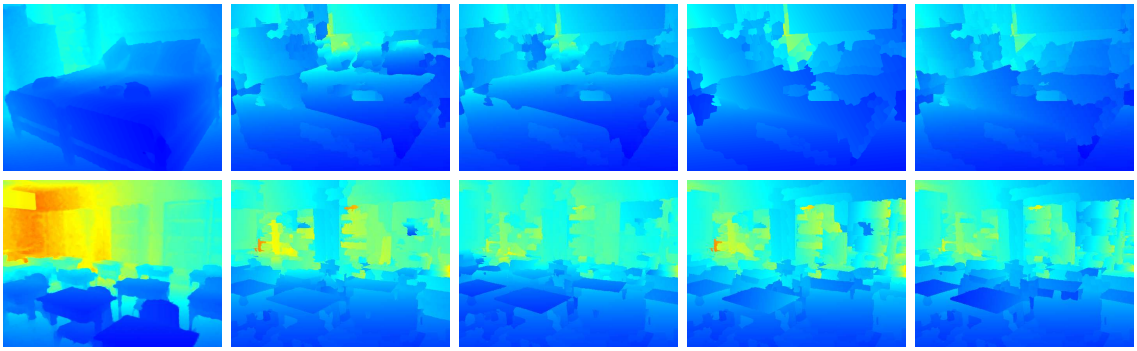


Figure 3.7: **NYUv2: Ablation study.** Depth maps obtained by the different components of our approach.

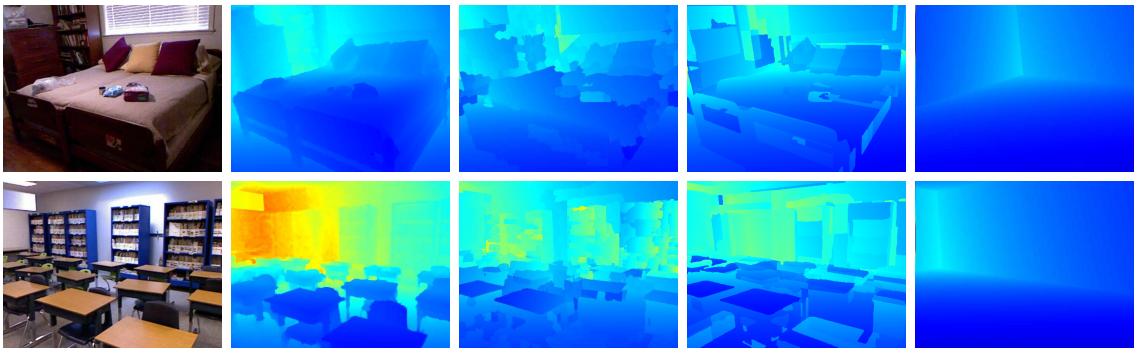


Figure 3.8: **NYUv2: Depth of the different layers in our model.** We show the depth maps estimated by our final model, corresponding to the variables associated with each layer in our hierarchy.

results of DeepDepth [Eigen et al., 2014] on the depth metrics are as follows. rel: 0.215; rms: 0.9; $\delta < 1.25$: 61.10%; $\delta < 1.25^2$: 88.70%; $\delta < 1.25^3$: 97.10%. While they are more accurate, recall that DeepDepth relies on a much larger training set. In terms of normal accuracy, shown in Table 3.2, we outperform the baselines on three out of the five metrics. It is worth noting that our model was not designed for surface

Method	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Ours-local	0.440	0.167	1.24	39.38%	72.41%	89.83%
Ours-mid	0.395	**0.159**	**1.22**	**41.25**%	**74.29**%	**90.75**%
Ours-global-only	0.423	0.167	1.26	38.64%	71.09%	88.76%
Ours	**0.379**	**0.159**	**1.22**	40.67%	73.67%	90.01%

Table 3.4: **RMRC Indoor: Ablation study.** We compare the different components of our approach. As with NYUv2, we observe that all the parts of our model contribute the its final result, with a large contribution from the mid-level structures.

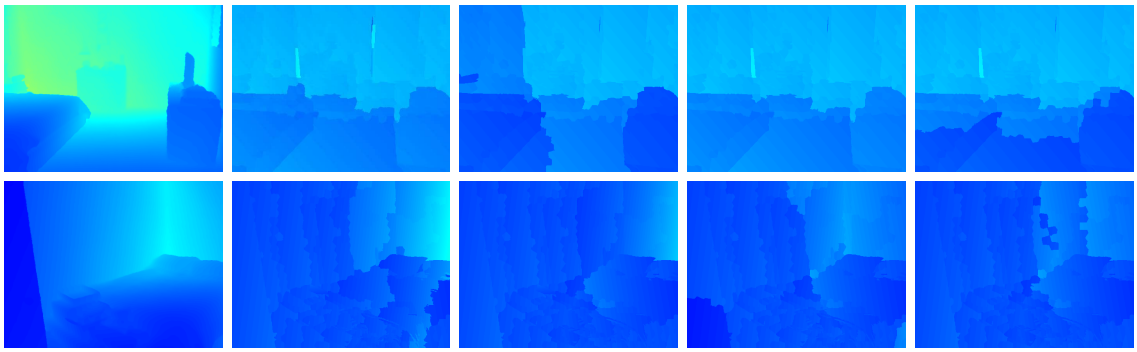


Figure 3.9: **RMRC Indoor: Ablation study.** Depth maps obtained by the different components of our approach.

normal estimation, which is a by-product of our depth estimation. Fig. 3.6 provides a qualitative comparison of the depth maps recovered by the different approaches on several images. Altogether, these results confirm that the use of mid-level and global structure is beneficial.

In Table 3.3, we provide an analysis of the different parts of our model. The analysis evidences the fact that each layer in the proposed hierarchy representation contributes to improving the final accuracy. It also reveals that the mid-level structures seem to yield the main improvement, among the two kinds of high-level structures. In Fig. 3.7, we provide a qualitative comparison of the depth maps obtained by the different components of our approach. Although not obvious at this scale, we observed that, while the mid-level structures help spatial depth coherence, they still respect the discontinuities in the image. Furthermore, the global structure yields more accurate depth ordering in the entire scene.

In addition to the depth of the superpixels, our model also predicts depth from the regions and from the global layout (although in a much coarser manner for the latter). Some resulting depth maps are depicted in Fig. 3.8.

Figure 3.10: **RMRC Indoor: Depth of the different layers in our model.** We show the depth maps estimated by our final model, corresponding to the variables associated with each layer in our hierarchy.

3.4.2 Evaluation on RMRC Indoor

We also evaluated our approach on the RMRC Indoor dataset [RMR, 2014]. Since this dataset does not provide ground-truth depth for the test images, and since our goal was to evaluate the different components of our model, we only employed the 4105 training images, from which we randomly sampled 114 images to form a test set with ground-truth. In this experiment, we used the same parameters as for NYUv2. In Table 3.4, we provide the various error metrics for the different parts of our model. As for NYUv2, we can see that each part contributes to the final results. Particularly, the influence of the mid-level structures seems to be even larger than on NYUv2. To provide the reader with a rough idea of how our results compare to other methods, it is worth mentioning that, on the test data of the RMRC Challenge [RMR, 2014], the best reported relative depth errors were 0.33 for [Eigen et al., 2014] and 0.39 for the second best approach of [Baig and Torresani, 2016]. In Figs. 3.9 and 3.10, we show the depth maps of the different components of our approach and the depth maps predicted by the variables in the different layers of our final model, respectively.

3.5 Conclusion

We have introduced a single image depth estimation approach that exploits the structure of the scene at different levels of details. Our experiments have demonstrated the benefits of such a structure-aware approach over local depth prediction methods. In particular, our evaluation has evidenced the fact that the mid-level structures, i.e., the regions, provided the largest contribution to the final accuracy of our model. A potential avenue for furture research would be to investigate if this can be leveraged to introduce better potentials in our model. Furthermore, incorporating the use of semantic labels in our depth prediction framework could also lead to potential improvement. Depth estimation attempts to predict absolute depth values, which could then potentially be exploited for other high-level scene understanding tasks.

Therefore, another potential research direction could be to jointly perform depth estimation and a related task, such as 3D box proposal generation, following the intuition that the related task can potentially benefit from the depth estimates. In the next chapter, we introduce our work on 3D box proposal generation that leverages depth estimation.

Chapter 4

3D Box Proposal from Monocular Image

Depth prediction, even though suffering from inaccuracy, has proven beneficial for other related computer vision tasks, such as semantic segmentation in [Eigen and Fergus, 2015]. In our work, we will utilize predicted depth maps to facilitate the tasks of 3D box object proposal in this chapter and scene parsing in chapter 5. In the following, we introduce our approach to generating 3D box proposals from a single monocular RGB image.

Modern object detection methods typically rely on bounding box proposals as input. While initially popularized in the 2D case, this idea has received increasing attention for 3D bounding boxes. Nevertheless, existing 3D box proposal techniques all assume having access to depth as input, which is unfortunately not always available in practice. However, depth estimation makes it possible to generate 3D box proposals from a single monocular RGB image. To this end, we develop an integrated, fully differentiable framework that inherently predicts a depth map, extracts a 3D volumetric scene representation and generates 3D object proposals, as illustrated in Figure 4.1. At the core of our approach lies a novel residual, differentiable truncated signed distance function module, which, accounting for the relatively low accuracy of the predicted depth map, extracts a 3D volumetric representation of the scene. Our experiments on the standard NYUv2 dataset demonstrate that our framework lets us generate high-quality 3D box proposals and that it outperforms the two-stage technique consisting of successively performing state-of-the-art depth prediction and depth-based 3D proposal generation.

In the previous chapter, we have performed depth estimation using traditional models, independent of deep learning, with dramatic improvement over previous methods also based on traditional techniques. Traditional models, however, have limited accuracy due to their lower learning capacity than deep networks. Therefore, in this chapter and the following one, we will leverage the progress of deep networks for depth prediction.

Below, we first introduce the motivation and our idea in Section 4.1, and review the related work in Section 4.2. We then describe our model in detail in Section 4.3, evaluate it in Section 4.4, and conclude this chapter in Section 4.5.

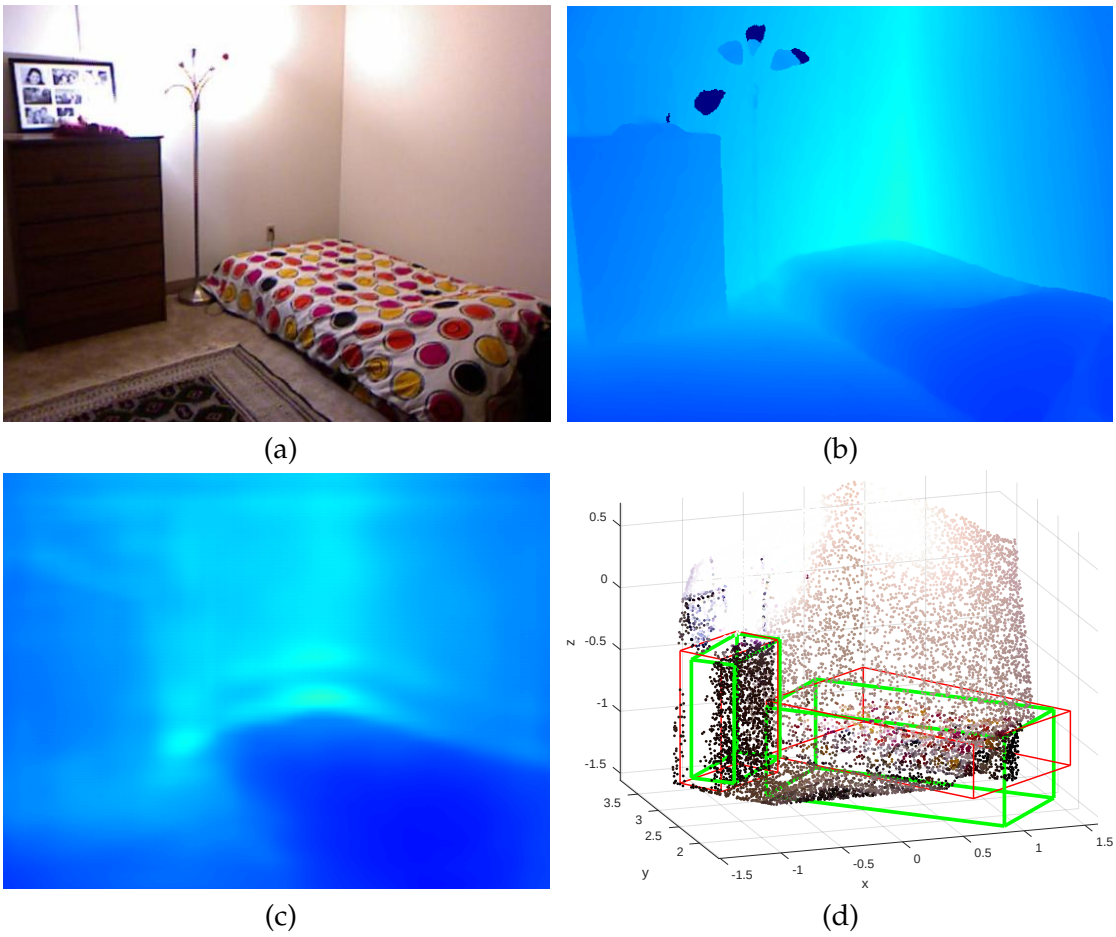


Figure 4.1: **3D object box proposal from a single image:** (a) Image; (b) ground-truth depth map; (c) Estimated depth; and (d) ground truth box(in red) and our proposals(in blue). Color indicates depth (red is far, blue is close).

4.1 Introduction

In the context of 2D scene understanding, generating class-independent object proposals, such as bounding boxes, has proven key to the success of modern object detectors; it has led not only to faster runtimes but also to more accurate detections [Ren et al., 2015]. Reasoning in 2D, however, only provides a limited description of the scene. A 3D interpretation would be highly beneficial for many tasks, such as autonomous navigation, robotics manipulation and Augmented Reality.

In recent years, several works have attempted to provide such a 3D interpretation by going beyond 2D bounding boxes. In particular, several methods have been proposed to model 3D objects with the 2D coordinates of the eight vertices of their 3D bounding box [Dwibedi et al., 2016; Hedau et al., 2010; Payet and Todorovic, 2011]. While this indeed better captures the shape of the object, e.g., by better adapting to orientations not parallel to the image axes, it still does not provide a 3D interpreta-

tion; each 3D bounding box can only be recovered up to scale. By contrast, [Fidler et al., 2012; Chen et al., 2016] truly reason in 3D. These works, however, have tackled the class-specific scenario in outdoor scenes, and would thus not generalize well to more cluttered environments, such as indoor scenes, and to arbitrary objects.

To the best of our knowledge, all the methods that consider the problem of generating class-independent object proposals [Chen et al., 2017; Song and Xiao, 2016] assume the availability of depth information. In particular, [Song and Xiao, 2016] achieved state-of-the-art results in indoor scenes by encoding a 3D scene with a truncated signed distance function (TSDF) and developing a region proposal network (RPN) based on 3D convolutions to generate proposals. In practice, however, depth is not always available, in which case these methods are inapplicable.

In this chapter, we therefore introduce an approach to generating class-independent 3D box proposals from a single monocular RGB image. Recent work [Song and Xiao, 2016] has shown a great advantage on proposal performance compared to previous approaches that fit 3D models to 2D predictions. Building upon this knowledge, we aim at learning proposals based on encoded 3D representations. Based on the recent progress in monocular depth estimation [Eigen and Fergus, 2015], the most straightforward way to doing so would be to rely on a state-of-the-art method to predict depth, followed by the state-of-the-art depth-based proposal generation technique of [Song and Xiao, 2016]. Here, however, we show that we can significantly outperform this two-stage approach by developing an integrated, fully-differentiable framework that can be trained in an end-to-end manner.

More specifically, we first propose a differentiable TSDF (DTSDF) module that can be appended to a depth-prediction network and produces an approximate TSDF-based representation. The quality of the resulting 3D representation, however, is limited by the accuracy of the predicted depth map and by our approximation of the TSDF, even when training the network end-to-end. To overcome this, we therefore introduce a residual version of our DTSDF module, which allows us to compensate for the depth inaccuracies and thus generate high-quality 3D box proposals.

We demonstrate the effectiveness of our method on the standard NYUv2 dataset. Our experiments evidence the benefits of the different components of our integrated framework. Furthermore, they show that our approach significantly outperforms the two-stage approach consisting of successive depth prediction and box proposal generation.

4.2 Related Work

Only limited work has been done on the task of generating class-independent 3D box proposals from a single monocular image. In this section, we review the methods that lie in the related fields of 2D and 3D box proposal generation, and the few existent methods that predict class-dependent boxes from monocular image.

Nowadays, the majority of object detection methods rely on generating class-independent object proposals. This trend was popularized in the 2D scenario, where

diverse proposal mechanisms have been developed [Uijlings et al., 2013; Cheng et al., 2014; Pinheiro et al., 2015; Hayder et al., 2016; Pont-Tuset et al., 2017], and has proven highly beneficial for both accuracy and runtime [Ren et al., 2015; Girshick, 2015]. In particular, in the current Deep Learning era, the region proposal network of [Ren et al., 2015], which shares feature computation across the different regions, has been adopted by several approaches [Song and Xiao, 2016; Dai et al., 2016; Hayder et al., 2017; He et al., 2017].

With the growing popularity of 3D scene understanding, it therefore seems natural that several works have turned to the problem of generating 3D box proposals. To this end, most approaches exploited the availability of depth sensors. In this setting, the main trend consists of fitting a 3D model to the given point cloud inside a 2D bounding box [Chen et al., 2017; Lin et al., 2013; Gupta et al., 2015; Deng and Latecki, 2017], which typically leads to class-dependent methods. By contrast, the work of [Maturana and Scherer, 2015; Song and Xiao, 2016] directly infers 3D boxes by learning shape features in a volumetric scene representation. There are also some other methods based on volumetric input. However, these methods focus on exploiting object-centric image or CAD models for object classification, and are nontrivial to extend to scene images [Wu et al., 2015; Qi et al., 2016]. In another research direction, the work of [Qi et al., 2017a,b] exploited symmetric representation of unordered 3D points. In this chapter, we introduce an approach that generates class-independent 3D box proposals directly from a single monocular image. While, in a different line of research, a few attempts have been made to learn a mapping from 2D images and 3D object representations [Fang et al., 2015; Wu et al., 2016; Girdhar et al., 2016; Zhou et al., 2017], these methods were developed for object-centric images, and are therefore not applicable to the complex indoor scene images that we deal with in this thesis.

A few methods have nonetheless also attempted to reason with 3D boxes from monocular input [Fidler et al., 2012; Chen et al., 2016]. These works, however, focused on the class-specific scenario, and have only tackled cases where a small number of classes are present in the scene. As such, they could not generalize to cluttered indoor scenes, and more importantly, to the problem of class-independent object proposal generation that we tackle here.

In short, to the best of our knowledge, our approach constitutes the first attempt at generating class-independent 3D box proposals from a single monocular image. To this end, we leverage the recent progress in monocular depth estimation [Karsch et al., 2012; Liu et al., 2014; Ladicky et al., 2014; Eigen et al., 2014; Zhuo et al., 2015; Eigen and Fergus, 2015; Liu et al., 2016; Laina et al., 2016] and develop a fully-differentiable residual module able to generate a volumetric representation of a cluttered indoor scene from an input image.

4.2.1 Deep Learning-based Depth Estimation

The first part of our framework consists of a deep convnet for depth estimation. Here, we briefly review the depth estimation methods, that are based on deep learning

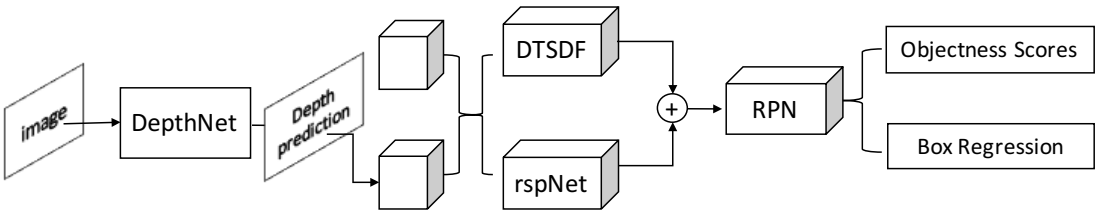


Figure 4.2: **Our 3D object proposal framework.** Our model consists of three parts integrated in a single architecture: a depth estimation network (DepthNet); a residual module to convert the predicted depth into a volumetric representation; a region proposal network (RPN). The middle part, which constitutes our key contribution, consists of a differentiable TSDF (DTSDF) encoding and of a residual-side-path network (rspNet) accounting for the predicted depth inaccuracies. These two subnetworks take two 3D grids as input, which correspond to a voxelization of the scene 3D volume and the projection of the depth map in this voxelization (see text for more detail). Ultimately, our model outputs the coordinates of 3D candidate boxes and corresponding objectness scores.

techniques.

Since the work of [Eigen et al., 2014], monocular depth estimation has entered the deep learning era. In contrast to traditional methods discussed in Chapter 3, deep models are usually trained with a much larger magnitude of data. [Eigen et al., 2014] utilized 120K image-depth pairs to train a deep network to predict the pixel-wise depth of a whole image. This network exploits the input at two scales: one branch is a typical VGG/Alex Net acting on low resolution input to capture global scene information; the other is a shallow convolution network taking as input of higher resolution image to retrieve details. [Eigen and Fergus, 2015] later extended this multi-scale deep convolutional neural network(DCNN) to three scales. Motivated by this work, many approaches exploited more complex deep frameworks for monocular depth prediction [Li et al., 2015; Wang et al., 2015a; Liu et al., 2015; Roy and Todorovic, 2016; Xu et al., 2017]. Recently, [Laina et al., 2016] improved the depth estimation accuracy by incorporating an efficient upsampling scheme based on residual learning. Instead of estimating absolute depth values, some works focused on its local difference, i.e., surface normal [Wang et al., 2015b; Bansal et al., 2016]. In summary, while these approaches improve the accuracy of predicted depth/surface normal for indoor scenes, they increase the complexity by incorporating stage-wise training and dependencies of related tasks. For simplicity and integrated training, we adopt the model of [Eigen and Fergus, 2015] to facilitate 3D box proposal generation from monocular image. In particular, deep learning based depth estimation played a critical role to estimate box parameters in real 3D space.

4.3 Methodology

As mentioned before, this work builds upon deep learning techniques, whose background is discussed in Chapter 2. Specifically, we aim to generate 3D object proposals from a single monocular image. To this end, we design a multi-task deep network that predicts a depth map, extracts a volumetric representation of the scene and generates 3D object proposals in the form of cuboids. The corresponding three subnetworks are shown in Figure 4.2. All three of them are differentiable, and the entire network can thus be trained in an end-to-end fashion. In the remainder of this section, we introduce our volumetric representation prediction network and the object proposal subnetwork, and then discuss our overall training strategy.

4.3.1 Volumetric Representation Prediction Network

To build a 3D volumetric scene representation, we first estimate a pixel-wise depth map from the input image and then compute an approximate TSDF from the corresponding point cloud. Below, we introduce our approach to addressing these two steps. Importantly, to be able to integrate the resulting modules in a complete multi-task network, we design them so that they are fully differentiable.

4.3.1.1 Depth Estimation

In this work, we adopt the VGG-based depth estimation network of [Eigen and Fergus, 2015] as our depth prediction network, whose detailed structure was introduced in Section 2.2.2, Chapter 2. In particular, we utilize the first two scales of the network of [Eigen and Fergus, 2015], which yields an output of size 55×74. We then upsample the resulting depth map to the full image size using bilinear interpolation, which can be cast as a convolution.

4.3.1.2 Differentiable TSDF

Given the depth map predicted by the depth network discussed above, we rely on a TSDF, introduced by [Newcombe et al., 2011], as our volumetric scene representation. In the accurate TSDF representation, the 3D space is divided into equally-spaced voxels. Each voxel is assigned a value encoding the distance of the voxel center to the closest surface point, derived from the depth map. Unfortunately, computing such an accurate TSDF is not differentiable with respect to the input due to the use of a nearest neighbor search procedure. To address this, and thus be able to exploit this for end-to-end training, we propose to make use of a differentiable, projective TSDF approximation. In particular, instead of looking for the nearest surface point in the entire 3D space, we only perform the search along the visual ray of each voxel, as exploited in [Newcombe et al., 2011; Song and Xiao, 2016]. Based on the projective TSDF, we introduce a soft truncation function to compute the final representation, which makes the entire process differentiable.

More formally, we divide the 3D space into $L \times H \times W$ equally-spaced cells of equal volume. Let us then denote by $G \in \mathbb{R}^{L \times H \times W}$ the 3D grid whose nodes encode the x, y, and z coordinates of the corresponding cell centers in the 3D world referential. Our goal now is to compute a TSDF value for each 3D point on G based on a projective approximation.

Under a perspective camera model, the image location $\mathbf{q} = [x_c, y_c]^T \in \mathbb{R}^2$ obtained by projection of a 3D point $\mathbf{p} = [x, y, z]^T \in \mathbb{R}^3$ can be expressed as

$$h(\mathbf{p}) = \lfloor \pi(KR^{-1}\mathbf{p}) \rfloor \,, \tag{4.1}$$

where π is the perspective projection function, that is, $\pi([x, y, z] = [x/z, y/z])$, $\lfloor \rfloor$ is the *floor* operator, and K and R are the matrix of camera intrinsic parameters and the camera rotation matrix, respectively. The latter only specifies the tilt angle, allowing us to align the scene according to the gravity direction. Furthermore, given a depth image $D \in \mathbb{R}^{M \times N}$, the 3D point $\hat{\mathbf{p}} = [\hat{x}, \hat{y}, \hat{z}]^T$ at image location $\mathbf{q}$ can be obtained as

$$\hat{\mathbf{p}} = c \cdot D(q), c = RK^{-1}[\mathbf{q}^T, 1]^T \,. \tag{4.2}$$

At each grid location $\mathbf{p}$ on G, the projective TSDF value can thus be obtained by projecting $\mathbf{p}$ and computing the distance between $\mathbf{p}$ the corresponding depth map point $\hat{\mathbf{p}}$.

More precisely, here, we consider the distances in the three directions x, y and z separately. Let $g(\mathbf{p}, \hat{\mathbf{p}}) = \mathbf{p} - \hat{\mathbf{p}}$, and $\psi = \|g(\mathbf{p}, \hat{\mathbf{p}})\|_2$ be the distance. The difference in z direction can be computed as $n(\hat{\mathbf{p}}) = [0, 0, 1]R^{-1}(\mathbf{p} - \hat{\mathbf{p}})$. Then, we define the truncated signed distance functions in x, y and z as

$$f(\mathbf{p}, \hat{\mathbf{p}}) = \begin{cases} \mathbf{1}\mu \cdot s(n(\hat{\mathbf{p}})) & \text{if } \psi/\delta \geq 1 \\ \min(|g(\mathbf{p}, \hat{\mathbf{p}})|, \mathbf{1}\mu) \cdot s(n(\hat{\mathbf{p}})) & \text{otherwise} \end{cases} \tag{4.3}$$

where $\mathbf{1} \in \mathbb{R}^3$ is a vector of ones, $|\cdot|$ computes the element-wise absolute value and $\min(\cdot)$ computes the element-wise minimum; $\delta = 0.1$, $\mu = 0.05$ and $s(x) = \tanh(k \cdot x)$ (with $k = 10$ in our experiments) truncates the distance to a signed constant. In words, $\hat{\mathbf{p}}$ encodes the surface points, and the sign of the distance then indicates whether the cell falls behind the surface, that is the cell is invisible (positive distance), or if it is visible (negative distance). We further assign zero values to the grid locations of G whose 2D projections fall out of the image range, which we refer to as invalid grid regions afterwards.

With this definition, the TSDF value at each location $\mathbf{p}$ on G is differentiable with respect to $\hat{\mathbf{p}}$. Therefore, following the chain rule, since the values of $\mathbf{p}$, K, R, and therefore $\mathbf{q}$ are fixed, the TSDF values are differentiable with respect to the depth prediction D. This will then allow us to employ this volumetric representation in an end-to-end learning framework.

	conv1	pool1	conv2	pool2	deconv1	deconv2
kernel	[3,3,3]	[2,2,2]	[3,3,3]	[2,2,2]	[3,3,3]	[3,3,3]
channels	3	3	3	3	3	3
stride	1	2	1	2	2	2

Table 4.1: **Parameters of our residual-side-path network.** The kernel size is represented in the order [width, length, height]. In the first convolution, we convert from the original 6 channels (3D cell center + 3D nearest point position along the visual ray) to 3. In the remaining layers, we maintain the number of channels to 3 to match the volumetric representation of the DTSDF and keep the memory requirement relatively low. For each layer, the strides are the same for all three dimensions. We do not use any biases in our layers.

4.3.1.3 Residual Network for Volumetric Representation

The DTSDF described above relies on the distance along the visual ray and, as mentioned before, is only an approximation to the true TSDF; the true nearest-neighbor to a 3D point might not be on its visual ray. We have found, however, that, in practice, the true neighbor is usually not far away from this approximation. Motivated by this observation, and to further account for the inaccuracies in the estimated depth map, we introduce a residual path to improve the DTSDF.

Specifically, the input to our residual-side-path network (rspNet) consists of a 3D grid similar to G defined above. However, instead of only encoding the 3D position of the corresponding cell center at each location, we further append to this position the coordinates of the corresponding surface point $\hat{\mathbf{p}}$ along the visual ray. This 3D grid with 6 channels then acts as input to an encoder-decoder network with the following structure

$$\begin{aligned} &input \rightarrow conv3d \rightarrow relu \rightarrow pool3d \rightarrow conv3d \rightarrow relu \rightarrow pool3d \\ &\rightarrow deconv3d \rightarrow relu \rightarrow deconv3d \rightarrow relu \rightarrow output \end{aligned} \tag{4.4}$$

where conv3d, deconv3d, pool3d represent convolution, deconvolution, pooling operations on a 3D grid, respectively. The parameters defining the layers of our rspNet are given in Table 4.1. Note that all convs/deconvs here contain no bias, so as to guarantee zero values in invalid grid regions.

Intuitively, the given input information allows the rspNet to compute the distance between the cell center and the corresponding surface point as in the DTSDF. However, by performing convolutions, it is also able to compensate for errors by looking at larger regions in the reconstructed volume and thus making use of context.

Altogether, our approach to volumetric representation prediction lets us effectively leverage the strengths of the explicit DTSDF computation and of the learning-based rspNet. While the explicit computation is less flexible, it provides with a reliable approximation of the true TSDF. By contrast, access to limited data might make it hard to use the rspNet on its own, but it provides more flexibility to compensate for the DTSDF and the depth prediction mistakes. The resulting volumetric

representation is then used as input to the region proposal network described below.

4.3.2 3D Object Proposal Generation

4.3.2.1 3D Object Proposal Network

We follow the recent trend in generating object proposals consisting of sharing feature computation, thus speeding up runtime [Ren et al., 2015; Song and Xiao, 2016]. In particular, since we work in 3D, we adopt the multi-scale 3D region proposal network (RPN) of the Deep Sliding Shapes (DSS) method of [Song and Xiao, 2016], whose detailed structure was introduced in Section 2.2.2, Chapter 2. This network relies on a volumetric representation as input, and is thus well suited to be appended to the framework discussed above. As output, it produces another volume at a lower resolution. To each voxel of the output volume are associated J anchors ($J = 19$ in our work). These anchors represent potential 3D object bounding boxes of different sizes and aspect ratios, and were defined according to the statistics of the training data. The RPN then predicts a probability for each anchor to correspond to an actual object at each cell location. Furthermore, it also regresses a 6D vector encoding the center position and the three side lengths of the corresponding 3D bounding box. Instead of pooling the features for each anchor separately, all the anchors of each voxel share their features, but have different classifiers/regressors.

4.3.2.2 Extended Anchors

In [Song and Xiao, 2016], runtime and accuracy were improved by removing empty anchors based on the input depth map. Here, however, we do not have access to the ground-truth depth maps, and our depth predictions are imperfect. In practice, we found that too many boxes were removed by this procedure. Furthermore, we also observed that our depth prediction often essentially differed from the true one by a single scale factor. Motivated by this, we therefore propose to enlarge the anchor pool by scaling the depth maps.

The range of the scale factors between predicted and true depth maps on the training set was found to be [0.8, 1.2]. We therefore scale the depth prediction with a global scale in this range with a stride 0.05. Assuming that all resulting scaled maps are valid ones, we only remove the anchors that do not contain any points of the scaled depth maps. Since this procedure can quickly lead to a huge number of anchors to consider, thus increasing runtime, we performed 3D non-maximum suppression to remove anchors with a large overlap. We further limited the number of valid anchors to 15,000 for each anchor type. Specifically, we keep all the non-empty anchors in the original, unscaled depth maps, and add anchors from the scaled depth maps by scoring them according to the corresponding inverse absolute scale difference with 1, e.g., $1/|1.2 - 1|$.

As evidenced by our experiments, the quality of the anchors is important to our results; our extended anchors allow us to obtain dense supervision in a huge 3D space during training, and, at test time, they prevent high-scoring proposals from not

being considered because they have been removed from the candidate pool. Nevertheless, as evidenced by our ablation study, the extended anchors are not the only key to the success of our approach; they rather help boosting the effectiveness of our residual volumetric prediction module.

4.3.3 Multi-task Loss and Network Training

We now turn to the problem of training our model. Assuming that we have access to ground-truth depth maps during training, we propose to define a multi-task loss consisting of two parts. The first one measures depth prediction error, and the second encodes errors on the generated proposals themselves. Specifically, we define our loss as

$$\mathcal{L} = \lambda \mathcal{L}_{depth}(D, D^*) + \sum_{i=1}^{N} \mathcal{L}_{rpn}(p_i, p_i^*, t_i, t_i^*) \ , \tag{4.5}$$

where $\mathcal{L}_{depth}$ is the depth loss between the predicted depth map D and the ground-truth one D^*, and $\mathcal{L}_{rpn}$ is the object proposal loss comparing the predicted class probability p_i and regressed box parameters t_i with the ground-truth ones p_i^* and t_i^* for each of the N candidate anchors.

For the depth loss, we adopt the same loss function as in [Eigen et al., 2014]. Let us denote the log difference of depth prediction with ground truth as $y_i = \log d_i - \log d_i^*$, where d presents depth on a pixel, the depth loss is defined as,

$$\begin{aligned} \mathcal{L}_{depth}(D, D^*) &= \frac{1}{n^2} \sum_{i,j} ((\log d_i - \log d_j) - (\log d_i^* - \log d_j^*))^2 \\ &= \frac{1}{n} \sum_i y_i^2 - \frac{1}{n^2} (\sum_i y_i)^2 = \frac{1}{n} \sum_i y_i^2 - \frac{1}{n^2} \sum_{i,j} y_i y_j \ . \end{aligned} \tag{4.6}$$

This depth loss respects the local depth variance by comparing relationships between pairs of pixels i, j in the output. In the last row of Eq. (4.6), the first term calculates an absolute-scale $l2$ error; when considering both terms, this loss calculates a scale-invariant error. In [Eigen et al., 2014], a relaxed form of Eq. (4.6), i.e., $\frac{1}{n} \sum_i y_i^2 - \frac{\nu}{n^2} (\sum_i y_i)^2$, $\nu = 0.5$, was used to generate good absolute-scale predictions with slightly quatitatively improved results. In our work, we use the same relaxed depth loss for training.

In practice, as evidenced by our experiments, we have found the depth loss to be important, as it ensures that the input to the DTSDF and to the rspNet remains meaningful for volumetric representation prediction.

The object proposal loss consists of two parts: a softmax loss for classification of object vs non-object and a smooth ℓ_1 loss on the regression variables. The 3D regression loss is a direct extension of the one commonly used in 2D [Ren and Sudderth, 2016; Girshick, 2015]. Assuming that the objects lie on the ground, a 3D bounding box can be defined by 7 parameters, $[X, Y, Z, L, H, W, \theta]$, where the first three are the coordinates of the box center, the following three are the side lengths in the

three directions, and θ is the orientation. In this paper, we approximate the orientation of each object by the global scene orientation, which can be estimated from the predicted depth [Uijlings et al., 2013]. We are therefore left with 6 parameters to estimate.

Let us denote by $[X^*, Y^*, Z^*, L^*, H^*, W^*]$, $[X, Y, Z, L, H, W]$, and $[X_a, Y_a, Z_a, L_a, H_a, W_a]$ the parameters of a ground-truth box, a predicted one, and an anchor, respectively. To keep the magnitudes of these different values more comparable, we make use of relative values defined as

$$\begin{aligned}
&t_x = \frac{(X - X_a)}{W_a}, t_y = \frac{(Y - Y_a)}{H_a}, t_z = \frac{(Z - Z_a)}{L_a} \\
&t_w = \log\left(\frac{W}{W_a}\right),\ t_h = \log\left(\frac{H}{H_a}\right),\ t_l = \log\left(\frac{L}{L_a}\right) \\
&t_x^* = \frac{(X^* - X_a)}{W_a}, t_y^* = \frac{(Y^* - Y_a)}{H_a}, t_z^* = \frac{(Z^* - Z_a)}{L_a} \\
&t_w^* = \log\left(\frac{W^*}{W_a}\right),\ t_h^* = \log\left(\frac{H^*}{H_a}\right),\ t_l^* = \log\left(\frac{L^*}{L_a}\right)\ ,
\end{aligned} \tag{4.7}$$

where $[t_x^*, t_y^*, t_z^*, t_w^*, t_h^*, t_l^*]$ denote ground-truth values and $[t_x, t_y, t_z, t_w, t_h, t_l]$ predicted ones.

Altogether, our object proposal loss can thus be written as

$$\begin{aligned}
&\mathcal{L}_{rpn}(p, p^*, t, t^*) = \mathcal{L}_{cls}(p, p^*) + \gamma \sum_{i \in s} r(t_i^* - t_i) \\
&w.r.t. \quad s = \{x, y, z, w, h, l\} \\
&\text{where} \\
&L_{cls}(p, p^*) = -p^* log(p) - (1 - p^*) log(1 - p) \\
&\text{and} \\
&r(x) = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}
\end{aligned} \tag{4.8}$$

is the smooth ℓ_1 loss.

In practice, we first generate depth maps using the depth network. We then remove the empty anchors, following the procedure used to generate the extended anchors discussed above, based on the depth predictions. At test time, we similarly remove the empty boxes based on the predicted depth maps, and further perform 3D non-maximum suppression according to the predicted probabilities p, with a threshold of 0.35 on the volumetric IoU. Furthermore, we only keep the top K proposals ($K = 150$ in our work) among the anchors of each of the 19 categories.

4.4 Experimental Evaluation

We evaluate our model on the NYUv2 dataset [Silberman et al., 2012a], which consists of RGB images with their corresponding depth maps. The ground-truth 3D bounding boxes are provided with the SUN RGB-D dataset [Song et al., 2015]. NYUv2 contains 795 training images and 654 test images. In our experiments, we use only RGB images as input, while ground-truth depth maps are employed for supervision during training. The initial VGG-based depth estimation network of [Eigen and Fergus, 2015] with only two scales achieves the following errors. rel: 0.141; $\log 10$: 0.060; $\delta < 1.25$: 77.65%; $\delta < 1.25^2$: 95.16%; $\delta < 1.25^3$: 98.81%, where the definitions of these metrics are introduced in Chapter 3. For the reconstruction volume, we adopt the range and resolution used in [Song and Xiao, 2016], that is, [-2.6, 2.6] meters horizontally, [-1.5,1] meters vertically, and [0.4, 5.6] meters in depth. Each cell has side lengths of 0.025 meters. Altogether this yields a volumetric representation of size $208 \times 100 \times 208$.

We implemented our model in tensorflow, and trained it on two NVIDIA Tesla P100, each with 16GB memory. We used mini-batches containing one image at each iteration. For each batch, we sampled negative anchors in a ratio of 1.2 w.r.t. the positive anchors, with the pos/neg labels assigned according to the rules of [Song and Xiao, 2016]. We set the initial learning rate to 0.001 and decreased it at a rate of 0.5 every 2 epochs. The final network was selected using a validation set of 50 images, which was taken out of the standard training set. In our experiments, we trained our model for at most 15 epochs. Training our deep network takes roughly 10 hours, and inference takes 2.72s per image on average.

4.4.1 Baselines

As mentioned before, this work constitutes the first attempt to tackle the problem of generating class-independent 3D box proposals from a single monocular image. Therefore, we developed our own baselines by making use of the state-of-the-art monocular depth estimation network of [Eigen and Fergus, 2015] (with all three scales, compared to two scales in our framework), followed by the state-of-the-art depth-based 3D proposal generation method of [Song and Xiao, 2016]. We further designed a baseline inspired by the effective faster R-CNN framework of [Ren et al., 2015]. In practice, we trained the baselines using mini-batches of the same size as ours, and an initial learning rate of 0.001. We discuss these baselines in more detail below.

4.4.1.1 Est-DSS

The original DSS framework [Song and Xiao, 2016] consists of a class-independent multi-scale region proposal network trained on the accurate TSDF representations of input depth maps. To work in our monocular setting, we replace the ground-truth depth maps with those predicted by the three scale model of [Eigen et al., 2014]. DSS relies on the accurate TSDF, computed from the true nearest neighbor of each voxel.

To better understand the accuracy loss incurred by relying on the projective TSDF in our model, we further developed another baseline, named **Est-DSS-Approx**, where we replaced the accurate TSDF with our approximate one within the DSS framework.

4.4.1.2 Est-Faster-RCNN

Another approach consists of directly predicting 3D bounding boxes from the 2D image. To develop a baseline that works in this setting, we made use of the state-of-the-art faster R-CNN 2D detection framework of [Ren et al., 2015]. More precisely, we modified this framework to regress 3D coordinates from 2D anchors. To nonetheless exploit depth information, we extracted HHA features [Gupta et al., 2014a] from the depth predictions. These features, in conjunction with color images, acted as input to train the proposal generation part of the faster R-CNN. Specifically, we placed 12 different 2D anchors at each node on the image grid. For each anchor, we then regress the depth of its 3D box center, the coordinates of the 2D projection of the box center, and its height, width and length in 3D space, relative to the anchor center and size, similarly to Eq. (4.7). As in our framework, the orientation of each box was estimated according to the global orientation of the scene.

In particular, let us denote by $[X^*, Y^*, Z^*, H^*, L^*, W^*]$, $[x^*, y^*]$, $[x_a, y_a, w_a, h_a]$ the parameters of a ground-truth box, 2D projections of the box center, and the parameters of a 2D anchor. Let f be the focal length of the camera. The regression targets, $[t_x^*, t_y^*, Z^*, t_h^*, t_w^*, t_l^*]$, are computed as

$$\begin{aligned} &t_x^* = \frac{x^* - x_a}{w_a}, \quad t_y^* = \frac{y^* - y_a}{h_a} \\ &t_w^* = log(\frac{W^*/Z^*}{w_a/f}), \quad t_h^* = log(\frac{H^*/Z^*}{h_a/f}), \quad t_l^* = log(\frac{L^*/Z^*}{h_a/f}) \end{aligned} \tag{4.9}$$

Let us denote by $[t_x, t_y, t_h, t_w, t_l, Z]$ the predictions, where Z is the predicted depth at 3D box center. We make use of the following loss function for training

$$\begin{aligned} &\mathcal{L}_{reg-faster-rcnn} = \mathcal{L}_{cls} + \gamma[(\log Z^* - \log Z)^2 + \sum_{i \in \hat{s}} r(t_i^* - t_i)] \\ &with \quad \hat{s} = \{x, y, h, w, l\} \ , \end{aligned} \tag{4.10}$$

where the first term is a classification loss similar to that in Eq. (4.8), and the remaining part is the regression loss.

4.4.2 Evaluation Metrics

We evaluate the accuracy of 3D object proposals by calculating their recall, according to a volume overlap with ground-truth larger than 0.25, and their average box overlap (ABO) with the ground-truth. Note that, in NYUv2, the ground-truth consists of 3D box parameters in the world referential with a tilt rotation calculated from the ground-truth depth. Since, in our monocular setting, we cannot have access to the

methods	bathtub	bed	bookshelf	box	chair	counter	desk
Est-Faster-RCNN	33.3	84.5	59.8	9.9	82.1	84.6	82.9
Est-DSS-Approx	70.8	94.8	44.8	10.6	83.8	**92.3**	79.4
Est-DSS	**75.0**	**95.5**	**49.4**	10.6	85.3	84.6	**84.4**
Ours	70.8	93.5	34.5	**15.6**	**87.4**	**92.3**	82.9
methods	dresser	door	nightstand	lamp	pillow	monitor	bin
Est-Faster-RCNN	76.4	4.9	70.8	34.5	27.3	8.3	45.8
Est-DSS-Approx	83.6	4.9	81.3	27.2	17.9	4.2	30.5
Est-DSS	**90.9**	**11.8**	79.2	**23.6**	24.1	0.00	**37.3**
Ours	83.6	5.9	**81.3**	20.0	**48.3**	**25.0**	**37.3**
	sink	sofa	table	tv	toilet	Recall	ABO
Est-Faster-RCNN	53.2	81.2	80.4	24.2	93.3	62.3	0.319
Est-DSS-Approx	71.4	90.6	89.7	21.2	**96.7**	63.6	0.346
Est-DSS	72.7	**92.0**	**91.8**	27.3	**96.7**	66.1	0.348
ours	**85.7**	89.9	89.7	**33.3**	93.3	**69.3**	**0.364**

Table 4.2: **Comparison of our model with the baselines on NYUv2.** We show the class-wise recalls, overall recall and ABO of the 2000 top scored 3D windows on test set. Note that our model outperforms the two-stage baselines and the faster R-CNN one in both overall recall and ABO, thus showing the benefits of having an end-to-end learning framework.

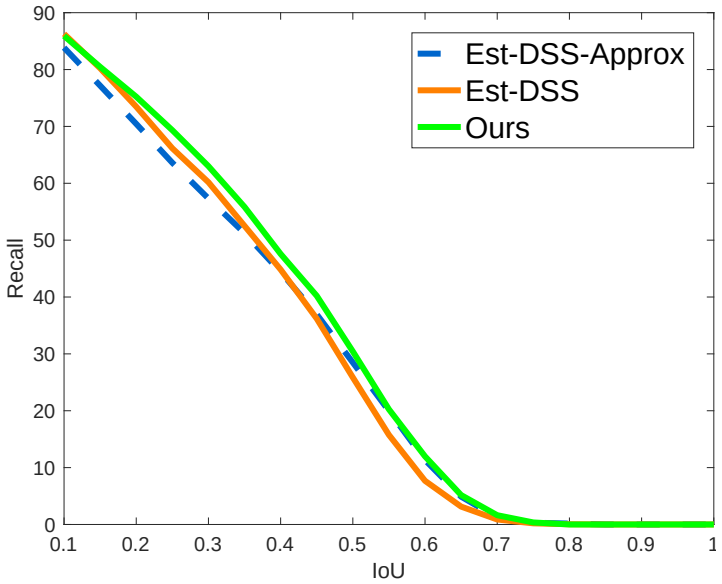


Figure 4.3: **Recall as a function of the IoU threshold.** Note that our approach outperforms the two-stage baselines, or performs on par with them, across the whole range of IoU thresholds, while the best performing baseline varies for low and high IoUs. This evidences the stability of our approach.

methods	recall	ABO	#Box
DSS	84.9	0.461	2000
3D Selective Search	74.2	0.409	2000
Ours	69.3	0.364	2000

Table 4.3: **Comparison to depth-based models on NYUv2.** We compare our model with methods based on ground-truth depth. Note that the gap between our model and these depth-based ones is relatively small, despite the fact that we rely only on a monocular image as input.

ground-truth tilt rotation, we estimate it using the initial depth estimates obtained from the network of [Eigen et al., 2014].

4.4.3 Experimental Results

We now present our results on the NYUv2 dataset. In Table 4.2, we first compare our complete model with the three baselines introduced above. For all methods, we selected the 2000 3D bounding boxes with highest score to calculate the recall and ABO. Our model significantly outperforms the baselines in terms of both recall and ABO, thus showing the importance of end-to-end training and the effectiveness of our residual volumetric prediction approach at compensating the errors in depth prediction and due to the TSDF approximation. Even though we tackle the problem of generating class-independent 3D object proposals, we also report the recall for each of the 19 object categories present in the dataset. Note that our model more effectively handles classes of small objects, such as sink, monitor and pillow, which are typically more challenging to detect. This, we believe, demonstrates that, while generic, our end-to-end training mechanism allows us to learn effective class-specific representations automatically. A qualitative comparison of our model with the best-performing Est-DSS baseline on a few images is provided in Figure 4.4.

Note also that exploiting volumetric representations, as done by Est-DSS, Est-DSS-Approx and our approach, seems to be more effective than direct regression to 3D as in our Est-Faster-RCNN baseline, which yields the least accurate proposals. We believe that this is due to the fact that the volumetric representation is better suited to capture the shape of 3D objects and their distances in 3D space. Finally, by comparing Est-DSS-Approx and Est-DSS, we can see that, as expected, the projective TSDF approximation yields worse results. Note, however, that our residual framework manages to compensate for this loss of accuracy, as better evidenced in the ablation study below.

In Figure 4.3, we plot the recall as a function of the IoU threshold for our method and the two-stage baselines. Note that we generally outperform, or perform on par with, the baselines on the whole range of IoU values. Note also that the best performing baseline differs for low and high IoU thresholds. This, we believe, further demonstrates the stability of our model.

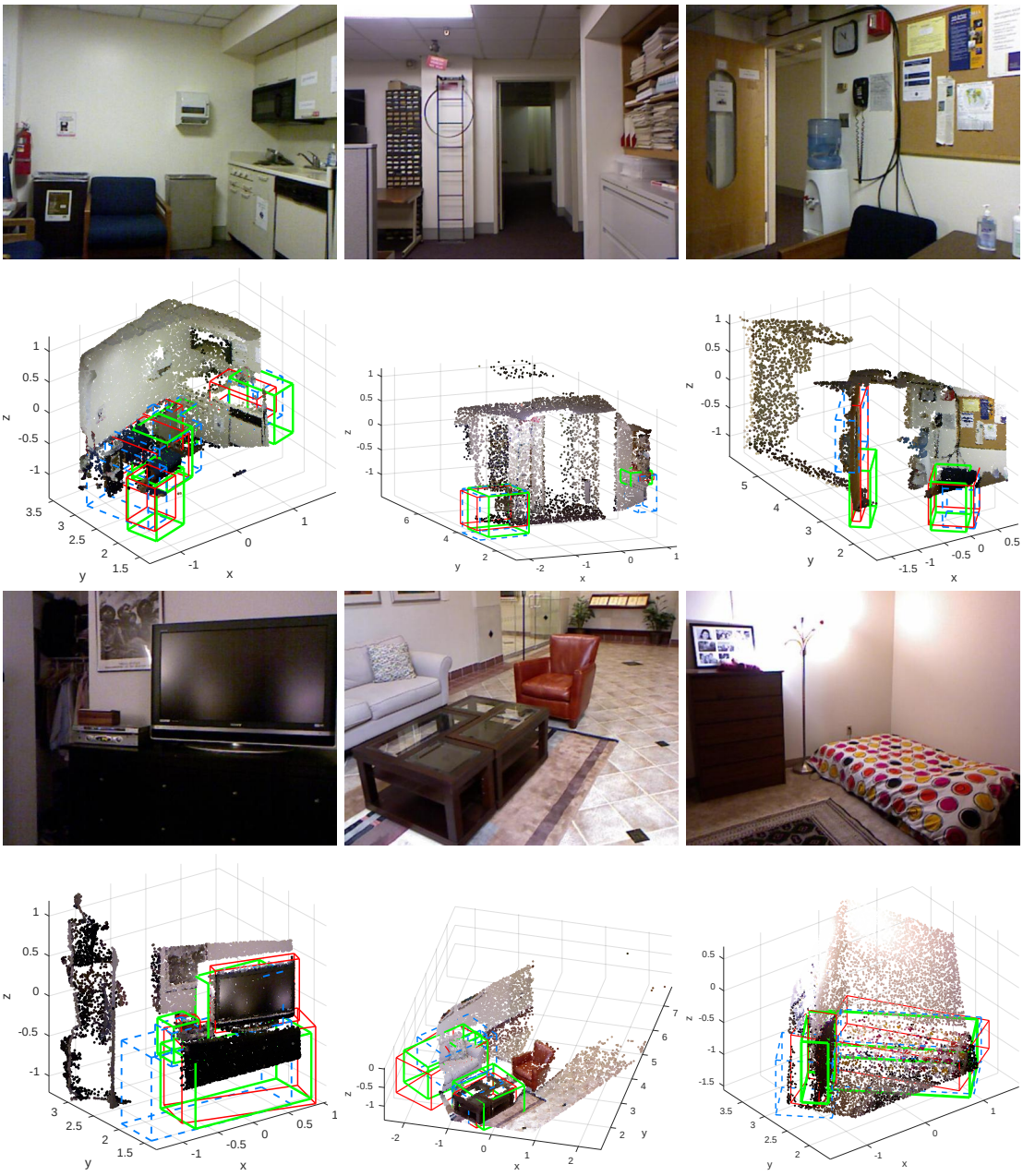


Figure 4.4: **Qualitative comparison of our model with Est-DSS on NYUv2.** We show the proposals with highest IoU returned by our model and by the baseline. The results of our model are shown in green, those of the baseline in dashed blue and the ground-truth boxes in red. Note that our proposals better match the ground-truth ones.

End-to-End	Acc.TSDF	Res. Path	Ext. Anchors	Depth Loss	Recall	ABO
					63.6	0.346
	✓				66.1	0.348
	✓		✓		65.7	0.356
✓		✓	✓		54.7	0.298
✓		✓		✓	66.3	0.360
✓			✓	✓	67.4	0.356
✓		✓	✓	✓	69.3	0.364

Table 4.4: **Ablation study on NYUv2.** We evaluate the influence of different components of our framework. Here, "End-to-End" indicates whether a model is end-to-end trained, and "Res. Path" indicates whether it has the residual path. Note that our residual volumetric prediction module is able to effectively compensate for our use of an approximate TSDF representation, and can be further improved by the use of our extended anchors, which, by contrast, have only little effect on the two-stage baseline.

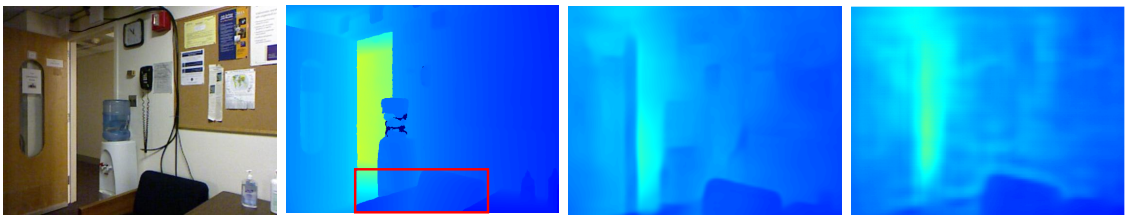
4.4.3.1 Comparison to Depth-based Models

In Table 4.3, we compare our results with those of depth-based models, i.e., DSS [Song and Xiao, 2016] and 3D selective search [Uijlings et al., 2013], whose results were obtained using the implementation by [Song and Xiao, 2016]. While our model, which relies only on an RGB image as input, yields slightly worse results that these methods that exploit ground-truth depth, the gap is remarkably small; e.g., we achieve only 4.9% lower recall than 3D selective search. Considering the strong ambiguities of depth estimation from a monocular image, we believe that this small gap shows the effectiveness of our monocular 3D box proposal model.

4.4.3.2 Ablation Study

In addition to the previous baseline comparison, we perform a comprehensive analysis of the impact of the different components of our approach. In particular, we evaluate the importance of (i) making our framework end-to-end trainable; (ii) relying on the accurate TSDF compared to the approximate one; (iii) our residual network for volumetric prediction; (iv) our extended anchors; (v) the use of the depth loss as intermediate supervision.

The results of this ablation study are provided in Table 4.4. In short, we can see that (i) our residual volumetric prediction is able to compensate for the loss in accuracy incurred by the use of the projective TSDF; (ii) Our extended anchors help further boost the accuracy of our model, while they only have little effect when used in conjunction with the two-stage baseline; (iii) depth supervision is important for our model, as it ensures that the input to our volumetric prediction remains meaningful. In our experiments, we set the weight of the depth loss λ to 1. We found, however, that our results were robust to this value, as long as it is sufficiently large. For example, with $\lambda = 10$, our model still outperforms the baseline with a



(a) (b) (c) (d)

Figure 4.5: **Example of predicted depth.** (a) input image; (b) ground-truth depth; (c) initial depth estimate [Eigen and Fergus, 2015]; and (d) depth estimate from our model. Even though our depth estimates are not globally more accurate than the initial ones, it better separates the foreground objects from the background, where we label the object with a red rectangle, as can be seen in (b).

recall of 67.4 and an ABO of 0.359. Interestingly, we have observed that our depth estimates, while not globally more accurate than the initial ones, better separate the foreground objects from the background, as in the example in Figure 4.5. This seems natural since we aim to generate proposals for the foreground objects. Altogether, we believe that this ablation study clearly evidences the strengths of the different components of our approach.

4.4.3.3 Generalization of our Model

To demonstrate the generality of our approach, we make use of the SUN-RGBD dataset [Song et al., 2015] to test our model and the Est-DSS baseline. The SUN-RGBD dataset consists of 5050 test images, including some from the NYUv2 dataset. For this evaluation to be more meaningful, we do not fine-tune the models, and explicitly exclude the NYUv2 images from the test set, thus leaving us with 4395 images. In practice, since the image size of SUN-RGBD changes, we simply resize them all to the size of the NYUv2 images, i.e., [427,561]. Furthermore, to adjust the camera intrinsic matrix to the new image size, we multiply the focal lengths and the principle point by the ratio of the NYUv2 image size to the original SUN-RGBD size of each input image.

The results of this experiment are provided in Table 4.5. Our model achieves a performance similar to that on the NYUv2 dataset. Furthermore, it outperforms the baseline in both overall recall and ABO, shows the ability of our model to generalize to new data. Specifically, we outperform the baseline on 10 out of the 19 categories in the dataset, with a large margin on objects of small sizes. The qualitative comparison of our model with the baseline is provided in Figure 4.6.

For completeness, we also report the results of our approach on the complete SUN-RGBD test set, including the images from NYUv2. This corresponds to a recall of 68.1% and an ABO of 0.354, both of which are slightly higher than our results on the previous SUN-RGBD subset.

methods	bathtub	bed	bookshelf	box	chair	counter	desk
Est-DSS	**75.0**	87.9	**51.6**	21.9	66.5	**76.5**	73.6
Ours	64.3	**89.5**	40.0	**24.1**	**72.1**	75.2	**77.3**
methods	dresser	door	nightstand	lamp	pillow	monitor	bin
Est-DSS	**69.6**	**11.2**	**60.0**	**24.5**	25.0	18.0	44.1
Ours	55.4	10.2	46.5	14.7	**34.1**	**27.0**	**51.8**
	sink	sofa	table	tv	toilet	Recall	ABO
Est-DSS	64.5	**84.3**	81.9	27.6	**90.0**	64.1	0.328
ours	**69.3**	82.1	**84.7**	**31.0**	85.0	**67.9**	**0.353**

Table 4.5: **Generalization study on SUN-RGBD excluding NYUv2.** We evaluate our model and a baseline model, both trained on NYUv2, on a subset of SUN-RGBD excluding the images from NYUv2. Our model remains more effective than the baseline on this data, thus evidencing the generality of our approach.

4.4.3.4 Failure cases

Even though our model performs dramatically better than the baselines, it still surfers inaccuracy due to depth estimation and the scene orientation estimation. Since we use the global orientation of scene, obtained by a pre-processing procedure, to model the rotations of all objects in the scene, the 3D box proposals can become inaccurate if the orientation is estimated poorly, or if the objects are not aligned to the room. Examples of failure cases are shown in Figure 4.7.

4.5 Conclusion

We have introduced an end-to-end method to generate class-independent 3D object proposals from a single monocular image. To the best of our knowledge, this constitutes the first attempt to work in this challenging setting for complex indoor scenes. Our experiments have demonstrated that our residual, fully-differentiable TSDF module produces an effective volumetric representation to generate box proposals, thus outperforming the two-stage approach based on the standard, non-differentiable TSDF. We have found that depth supervision was beneficial to our model. Importantly, however, our model does not require accurate depth on all parts of the image. In particular, the accuracy of the background depth is unimportant since we focus on foreground objects. For future work, it could therefore be interesting to modify the depth loss to focus more strongly on the foreground objects, and to estimate the orientation of each object individually.

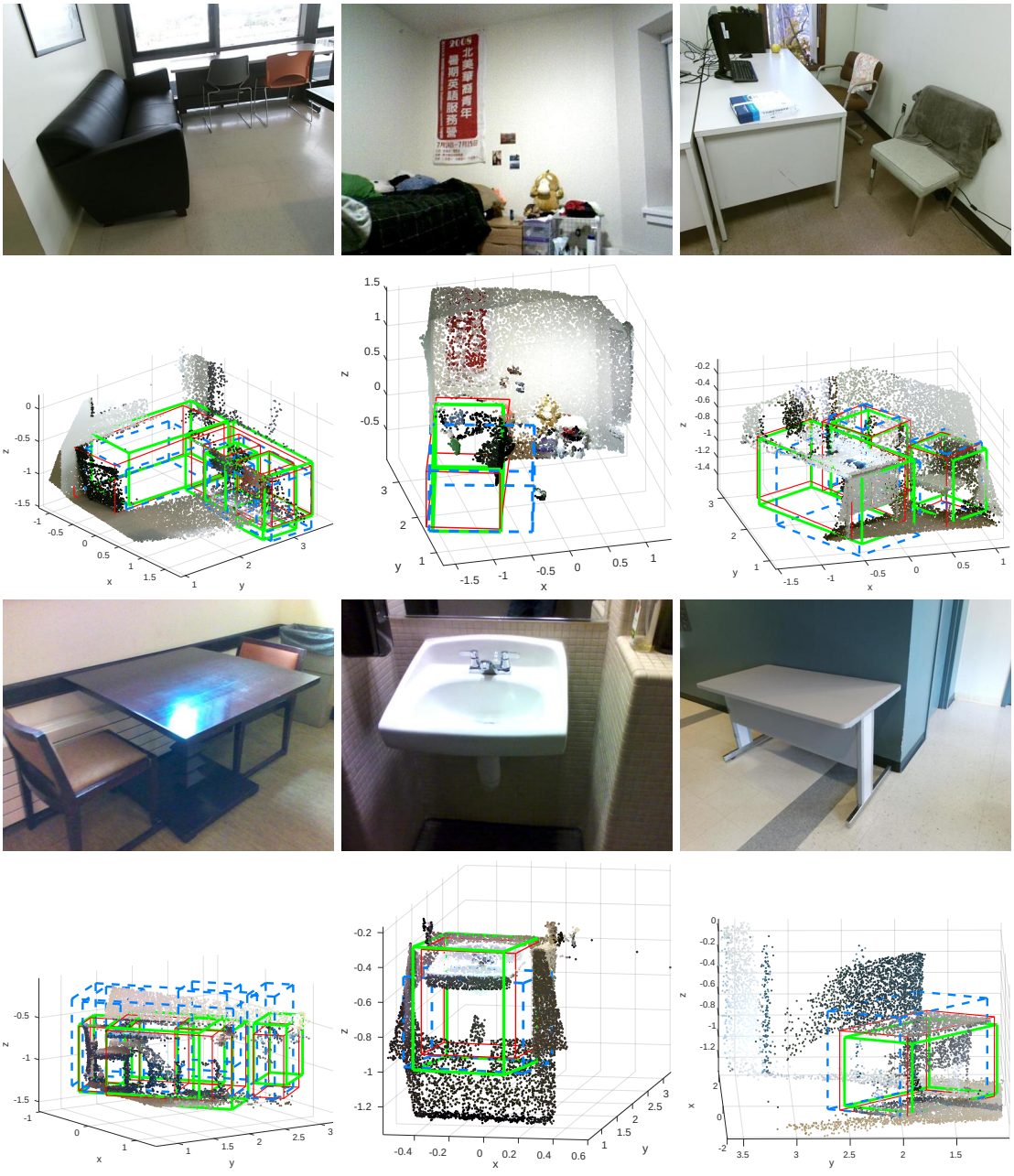


Figure 4.6: **Qualitative comparison of our model with Est-DSS on SUN-RGBD excluding NYUv2.** We show the proposals with highest IoU returned by our model and by the baseline. The results of our model are shown in green, those of the baseline in dashed blue and the ground-truth boxes in red. In addition, we rotate the scene for better visualization of these 3D boxes.

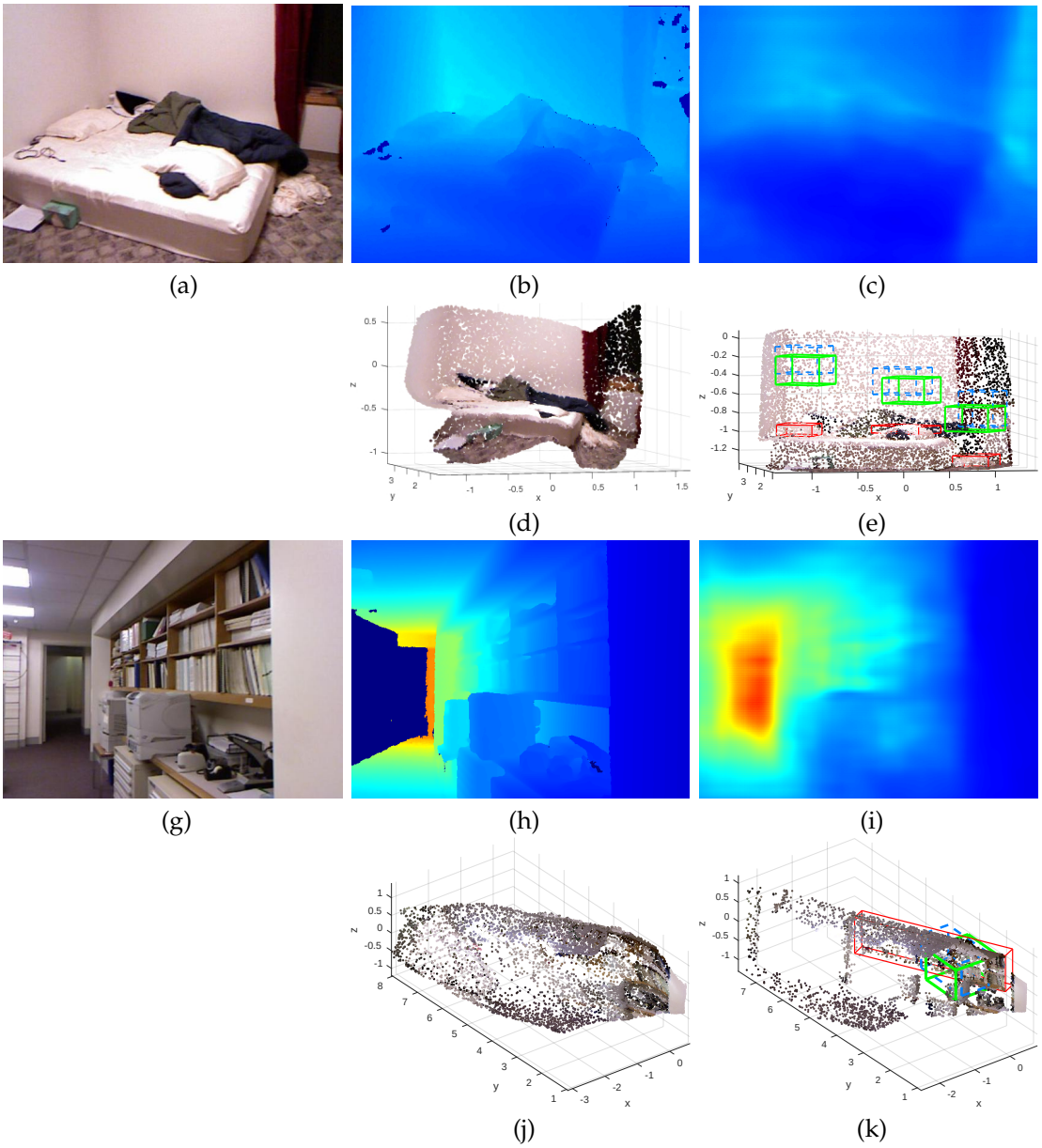


Figure 4.7: **Failure examples.** (a),(g) input images; (b),(h) ground-truth depth; (c),(i) estimated depth; (d),(j) estimated 3D reconstruction; and (e),(k) proposals of our method (green), the baseline (blue), and ground-truth (red). In the first example, because of the similar appearance of the foreground (pillow) and the background (bed), depth cannot be predicted accurately. Inaccurate depth estimation leads to the failure of our method as well as of the baseline. In the second example, the inaccurate estimation of the global scene orientation, used for box orientations, leads to inaccurate 3D box proposals.

Chapter 5

Indoor Scene Parsing with Instances, Semantics and their Supports

In the previous chapters, we analyzed scenes by estimating detailed depth maps, and 3D object boxes. For a comprehensive scene representation and understanding, in this chapter, we parse a scene into meaningful regions corresponding to instances, estimating their semantics, and predicting their support relationships. We perform these tasks based on a single monocular image, and, as in Chapter 4, make use of depth predictions. Depth predictions provide important information for predicting support relationships, and improving instance segmentation over previous methods which are blind to this information. In practice, these tasks have many applications. In particular, instances and semantics can facilitate a computer or robot to better understand a scene or the surrounding environment, and in the end to implement its ultimate target, e.g, navigation. The support relationships provide important information for a robot to grasp and place objects.

For the scene parsing task mentioned above, we propose an integrated model to solve the three subtasks, i.e., instance segmentation, semantic labeling, and support relationship inference simultaneously. Existing methods have typically focused on diverse subtasks of this challenging problem. In particular, while some of them aim at segmenting the image into regions, such as object instances, others aim at inferring the semantic labels of given regions, or their support relationships. These different tasks are typically treated as separate ones. However, they bear strong connections: good regions should respect the semantic labels; support can only be defined for meaningful regions; support relationships strongly depend on semantics. In this chapter, we therefore introduce an approach to jointly learn the three sub-tasks. By exploiting a hierarchical segmentation, we formulate our problem as that of jointly finding the regions in the hierarchy that correspond to instances and estimating their class labels and pairwise support relationships. We express this via a Markov Random Field, which allows us to further encode links between the different types of variables. Inference in this model can be done via integer linear programming, and we learn its parameters in a structural SVM framework.

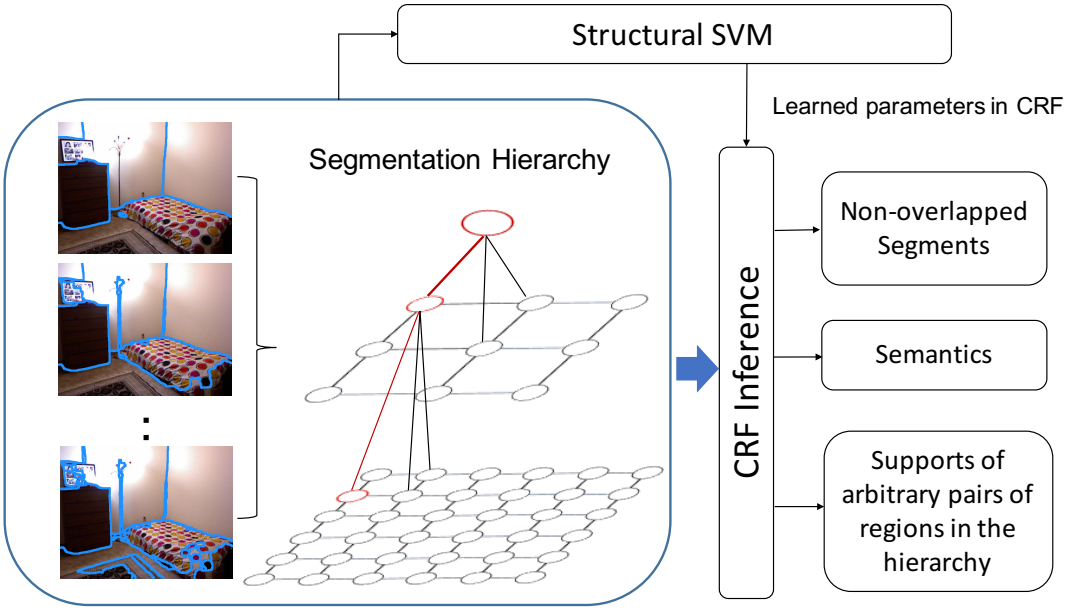


Figure 5.1: Our scene parsing framework for instance segmentation, semantic labeling and support relationship inference based on a segment hierarchy. Each node in the hierarchy graph indicates a region/segment, and an edge connecting two nodes in different layers indicates that one region includes the other.

Below, we first introduce the motivation and our idea in Section 5.1, and review the related work in Section 5.2. We then describe our model in detail in Section 5.3, and evaluate it in Section 5.4. We conclude this chapter in Section 5.5.

5.1 Introduction

Indoor scene understanding is one of the core challenges in computer vision. It aims at providing detailed information about the objects in a scene, such as their type and how they interact with each other. Such a level of understanding could have a high impact in many applications, such as personal robotics, where, to be able to interact with objects, one needs to reason about their semantics and how they are placed relative to each other.

In essence, indoor scene parsing is a complex problem that consists of multiple subtasks, such as segmenting the scene into meaningful regions [Arbeláez et al., 2014; Gupta et al., 2013; Shi and Malik, 2000], e.g., object instances, predicting semantic labels for every pixel in the scene [Long et al., 2015; Eigen and Fergus, 2015; Zheng et al., 2015] and reasoning about the support relationships of different regions [Jia et al., 2013; Guo and Hoiem, 2013; Silberman et al., 2012a; Yang et al., 2017]. In the literature, with the exception of [Silberman et al., 2014] that jointly reasons about regions and semantics, existing approaches typically tackle these subtasks independently. These subtasks, however, truly are strongly connected. For instance, the support relationship of two regions is highly correlated with their semantics; reason-

ing about support can be facilitated by using semantically meaningful regions. By addressing these tasks separately, or sequentially, existing methods cannot leverage the full collective power of all these dependencies.

In this chapter, we therefore introduce an approach to jointly segment the object instances and infer their semantic labels and support relationships in an indoor scene from a single input image. To this end, we exploit a hierarchical segmentation and formulate our problem as that of finding the regions corresponding to instances in this hierarchy, while simultaneously predicting a semantic label for each such region and the support relationship between any pair of such regions. We jointly express these subtasks in a single Conditional Random Field (CRF). This allows us to effectively encode the dependencies between them, thus leveraging all the connections underlying our overall problem.

We perform inference in the resulting CRF by formulating it as an integer linear programming problem. To cope with the size of this problem, we propose to make use of a regressor trained to predict the overlap of each region with a ground-truth object instance to effectively prune the region candidates. Thanks to the efficiency of this reduced inference strategy, we can learn the parameters of our model using structural Support Vector Machines (SVM). To this end, we design a loss function that reflects the multi-task nature of our indoor scene parsing formalism. Our framework is illustrated in Figure 5.1.

We demonstrate the effectiveness of our approach on the NYUv2 dataset [Silberman et al., 2012a]. Our experiments evidence that accounting for the dependencies between regions, their semantics and their support helps improving the prediction of the corresponding variables, with a particularly high impact on support relationships.

5.2 Related Work

Indoor scene understanding has been an important research focus in the computer vision community. As discussed above, this challenging problem consists of multiple subtasks. In particular, here, we tackle the tasks of instance segmentation, semantic labeling and support relationship prediction. We therefore focus on reviewing work in the three sub-categories.

5.2.1 Segmentation of Instances and Semantics

Segmenting an image into regions has attracted a huge interest over the years [Arbelaez et al., 2011; Arbeláez et al., 2014; Comaniciu and Meer, 2002; Shi and Malik, 2000], and often acts as a pre-processing step for some high-level recognition tasks. A complete review of this literature goes beyond the scope of this chapter. Here, we briefly discuss the methods that have been used for indoor scene understanding. In this context, the most direct approach consists of using standard over-segmentation methods, such as SLIC [Achanta et al., 2012], Mean-Shift [Comaniciu and Meer, 2002]

and normalized-cut [Shi and Malik, 2000]. In [Ladickỳ et al., 2014], multiple over-segmentations were employed jointly for monocular normal estimation. By contrast, many approaches favor exploiting hierarchical segmentations [Arbelaez et al., 2011; Arbeláez et al., 2014; Gupta et al., 2013; Hoiem et al., 2011]. While some works then select specific levels in this hierarchy, such as [Ren et al., 2012] and our work in Chapter 3, others aim to automatically find the best active regions in it, e.g., that fit the image contours [Hoiem et al., 2011], or whose pixel intensities follow a Gaussian distribution [Hu et al., 2015].

In particular, for instance segmentation, it is straightforward to build a model selecting regions that best match instances in the hierarchical segmentation [Silberman et al., 2014]. In an orthogonal research line, some work learned detailed instance directly using deep neural networks [Dai et al., 2016; Ren and Zemel, 2017].

Traditionally, semantic segmentation methods also often relied on pre-defined image regions [Ren et al., 2012; Silberman et al., 2012a; Gupta et al., 2013; Tighe and Lazebnik, 2010]. The motivation behind this was both computational cost and robustness to noise. Indeed, early approaches to semantic segmentation often relied on CRFs, in which inference can be computationally expensive when working at pixel level. Furthermore, working with regions allows one to regularize the predictions spatially. With the recent advent of deep learning, and progress in efficient inference methods [Koltun, 2011], many approaches now work directly at the level of pixels [Long et al., 2015; Eigen and Fergus, 2015; Zheng et al., 2015; Zhao et al., 2017]. [Long et al., 2015] introduced a fully convolution neural network (FCNN) which replaced the fully connected layers of typical DCNN models with convolutional layers and a bilinear upsampling, and thus enabling pixel-wise prediction independent of the input resolution. [Zheng et al., 2015] stacked this FCNN with a recurrent neural network approximating the conditional random field (CRF) inference. CRF explicitly models long-range links within a scene, which can sharpen the semantic boundary and smooth the predictions within a segment. A contemporary work [Chen et al., 2015] proposed an efficient deep structured learning strategy that unified the learning of CNN features and a graphical model. Motivated by this idea, some work stacked multiple CRF on different scales of features [Lin et al., 2016]. In the context of DCNN, [Zhao et al., 2017] achieved impressive performance on scene segmentation by introducing the technique of pyramid pooling, consisting of spatial poolings with large, diverse strides, that effectively extracts the global contextual information.

5.2.2 Region Relationship Inference

By contrast, when it comes to estimating support relationships, the notion of regions remains necessary. The task of estimating support was first introduced in [Silberman et al., 2012a], where a hierarchical segmentation was used to predict two support relationships: support from below, or from behind between pairs of regions. In this context, [Guo and Hoiem, 2013] predicts the height and extent of surfaces that can support objects or people. In [Jia et al., 2013] , instead of 2D segments, support is defined between 3D boxes. More recently, [Yang et al., 2017] proposed to make use

of object classes and physical stability to reason about support relationships between regions. All these methods make use of an RGBD image as input. By contrast, in this chapter, we aim to predict support from a single, standard RGB image.

More importantly, most of the methods discussed above tackle a single subtask of the challenging indoor scene understanding problem. The only two exceptions we are aware of are [Silberman et al., 2014], which jointly selects active regions in a hierarchy and predicts their semantic label; and [Silberman et al., 2012a], which jointly reasons about semantics and support relationships. Both of these works, however, also make use of RGBD as input. By contrast, we aim to jointly segment the object instances and infer their semantics and their support relationships from a single RGB image. To the best of our knowledge, our work constitutes the first attempt at considering all three subtasks together.

5.3 Methodology

Our goal is to jointly solve three sub-problems of indoor scene understanding, i.e., instance segmentation, semantic labeling and support relationship prediction, so as to account for their dependencies. To this end, we make use of a segmentation hierarchy, obtained by the method of [Gupta et al., 2013]. Our problem then translates to that of selecting the regions that best match ground-truth instances in this hierarchy, predicting their semantic label and their pairwise support relationships. We express this as inference in a CRF with three types of nodes: region selection ones, semantic label ones and support relationships ones. The edges in the model encode the dependencies between these variables.

More specifically, let us assume to be given a hierarchy of R regions forming a tree. To select the active regions in this tree, we define a set of binary variables $A = \{a_i\}_{i=1}^{R}$, $a_i \in \{0,1\}$. Furthermore, let $M = \{M_i\}_{i=1}^{R}$, $M_i \in \{1,\dots,K\}$ be the set of semantic labeling variables defining the class to which a region belongs, for K semantic classes. We then define an additional set of variables to model the support relationships between any two regions. To this end, let S_{ij} denote the type of support that region j provides to region i. Following [Silberman et al., 2012a], we consider three different cases: No support ($S_{ij} = 0$); j supports i from below ($S_{ij} = 1$); j supports i from behind ($S_{ij} = 2$). Note that we will often refer jointly to the latter two types as positive support, as opposed to the first type that corresponds to negative support. Furthermore, we introduce a hidden region to model the fact that some regions may be supported by a region that is not visible in the image. Altogether, the support variables can be expressed as $S = \{S_{ij}\}_{i=1,j=0}^{R}$, $S_{ij} \in \{0,1,2\}$, where $j = 0$ corresponds to support by the hidden region.

We then formulate the problem of jointly inferring these three types of variables

as that of maximizing the function

$$E(A,M,S) = \sum_{i=1}^{R} \phi_a(a_i) + \sum_{i=1}^{R} \phi_{ma}(M_i,a_i) + \phi_{tree}(A) + \sum_{i=1}^{R}\sum_{j=0}^{R} \phi_s(S_{ij}) + \sum_{i=1}^{R}\sum_{j=0}^{R} \phi_{sa}(S_{ij},a_i,a_j) \tag{5.1}$$

with respect to A, M and S, which can equivalently be converted to minimizing a CRF energy. The function relies on several potentials, which we discuss below.

The first term $\phi_r(a_i)$ is a unary potential encoding the probability that region i is active. We define this potential as $\phi_r(a_i) = w_a^T f_i^a [a_i = 1]$, where $[\cdot]$ is the indicator function, thus setting this potential to zero when $a_i = 0$. The vector f_i^a is a feature vector defined in Section 5.3.3, and w_a is the corresponding parameter vector to be learned from data.

The potential $\phi_{ma}(M_i,a_i)$ encodes the probability of predicting a particular semantic label for region i if the region is active. Simultaneously, it assigns a fixed cost to inactive regions. This can be expressed as

$$\phi_{ma}(M_i,a_i) = \begin{cases} 0 & a_i = 0 \\ w_{ma:M_i}^T f_i^{ma} & a_i = 1 \end{cases} \tag{5.2}$$

where f_i^{ma} is a feature vector, which, as described in Section 5.3.3, links semantics and support relationships. The vector $w_{ma:M_i}$ contains the parameters corresponding to each class M_i and will be learned from data.

The potential $\phi_{tree}(A)$ enforces constraints on the set of active regions. For the segmentation to be valid, every pixel in the image should be covered by a single region. This is achieved by making sure that only one region is selected in every path from the root of the segmentation hierarchy to a leaf node, such as the path in red in Figure 5.1. To this end, we thus define $\phi_{tree}(A) = \sum_{\gamma\in\Gamma} -\infty[1 \neq \sum_{i\in\gamma}[a_i = 1]]$, where Γ is the set of all root-to-leaf paths in the tree.

The unary potential $\phi_s(S_{ij})$ encodes the probability of a support variable to belong to either of the three classes. We write this potential as

$$\phi_s(S_{ij}) = w_{s:S_{ij}}^T f_{ij}^s \,, \tag{5.3}$$

where f_{ij}^s is a feature vector, which, as described in Section 5.3.3, links support types and semantics. The parameter vector $w_{s:S_{ij}}$ for each class S_{ij} will also be learned.

Finally, $\phi_{sa}(S_{ij},a_i,a_j)$ is a higher-order potential encoding the dependencies between the support variables and the region selection ones. We define this potential

as

$$\phi_{sa}(S_{ij}, a_i, a_j) = w_{sa} \begin{cases} w_b^T f_{ij}^{sa}, & S_{ij} \neq 0 \wedge (a_i = 0 \vee a_j = 0)] \\ w_c^T f_{ij}^{sa}, & S_{ij} \neq 0, a_i = 1, a_j = 1 \\ 0, & otherwise\ , \end{cases} \tag{5.4}$$

where f_{ij}^{sa} is a feature vector on a pair of regions, as described in Section 5.3.3. The vector w_b contains the parameters corresponding to the scenario where we predict a positive relationships even though either region is inactive, and w_c is the parameter vector for the case where both regions are active and we predict a positive relationship. Typically, we would like to penalize the first case and favor the second one. Other cases are assigned a fixed cost of zero.

5.3.1 Inference

To perform the inference in our model, we propose to re-write it as an integer linear program (ILP). To this end, let $a \in \mathcal{B}^{2R+1}$ be a vector of binary variables representing the states of A, where $a_{i,1} = 1$ encodes the fact that region i is active, while $a_{i,0} = 1$ corresponds to an inactive region i. Here, we add an extra variable $a_{0,1} = 1$ corresponding to the hidden region and forcing it to always be active. Furthermore, $m = \{m_{i,k}\}$, $1 \leq i \leq R$, $0 \leq k \leq K$, denotes binary variables encoding the pairwise state space of M and A, where $m_{i,0}$ represents the case where $a_i = 0$ for an arbitrary M_i, and $m_{i,k\neq 0}$ corresponds to the pairwise state $a_i = 1$ and $M_i = k$. Additionally, let $s = \{s_{i,j,t\in\{0,1,2\}}\}$ encode the state of the support relationship variables, and z the triplet states corresponding to the higher-order term $\phi_{sa}(S_{ij}, a_i, a_j)$, where $z_{i,j,l}$, $l \in \{1, 2, 3\}$, corresponds to the three cases in Eq. (5.4).

Inference in our model can then be re-written as the binary linear program

$$\begin{aligned} \underset{a,m,s,z}{\text{argmax}} \quad & \sum_{i=1}^{R} \theta_i^a a_{i,1} + \sum_{i=1}^{R}\sum_{k=0}^{K} \theta_{i,k}^m m_{i,k} + \\ & \sum_{i=1}^{R}\sum_{j=0}^{R}\sum_{t=0}^{2} \theta_{i,j,t}^s s_{i,j,t} + \sum_{i=1}^{R}\sum_{j=0}^{R}\sum_{l=1}^{3} \theta_{i,j,k}^{sa} z_{i,j,l} \end{aligned} \tag{5.5}$$

subject to

$$a_{i,l},\ m_{i,u},\ s_{i,j,t}\ z_{i,j,l} \in \{0,1\} \quad \forall i, l, j, t, u, v \tag{5.6}$$

$$a_{0,1} = 1\ , \tag{5.7}$$

$$a_{i,0} + a_{i,1} = 1, \quad \forall i \tag{5.8}$$

$$\textstyle\sum_{k=0}^{K} m_{i,k} = 1, \quad \forall i \tag{5.9}$$

$$m_{i,0} = a_{i,0}, \quad \forall i \tag{5.10}$$

$$\sum_{t=0}^{2} s_{i,j,t} = 1, \quad \forall i,j \tag{5.11}$$
$$\sum_{l=1}^{3} z_{i,j,l} = 1, \quad \forall i,j \tag{5.12}$$
$$\sum_{i \in \gamma} a_{i,1} = 1, \quad \forall \gamma \in \Gamma \tag{5.13}$$
$$\sum_{t \in \{1,2\}} \sum_{j=0}^{R} s_{i,j,t} \geq a_{i,1}, \quad \forall i \tag{5.14}$$
$$\sum_{t \in \{1,2\}} (s_{i,0,t} + s_{i,j,t}) \leq a_{i,1}, \quad \forall i, j \neq 0 \tag{5.15}$$
$$s_{i,0,1} \geq m_{i,1}, \quad \forall i \tag{5.16}$$
$$z_{i,j,2} = s_{i,j,0}, \quad \forall i,j \tag{5.17}$$
$$z_{i,j,3} \leq \sum_{t=1}^{2} s_{i,j,t}, \quad \forall i,j \tag{5.18}$$
$$z_{i,j,3} \leq a_{i,1}, \quad \forall i,j \tag{5.19}$$
$$z_{i,j,3} \leq a_{j,1}, \quad \forall i,j \tag{5.20}$$
$$z_{i,j,3} \geq \sum_{t=1}^{2} s_{i,j,t} + a_{i,1} + a_{j,1} - 2, \quad \forall i,j, \tag{5.21}$$

where the $\theta_\cdot$s encode the different potentials described above. The constraints can be interpreted as follows: Eqs. (5.7) – (5.12) enforce the binary variables to correspond to valid predictions. Eq. (5.13) enforces the tree constraints on the region selection variables. Eq. (5.14) forces a region to be supported by at least one region when it is active. This constraint encodes the fact that there is no floating region in the real world. Eq. (5.15) prevents a region to be supported by the hidden region if there is a region in the scene that can support it. Eq. (5.16) forces a region to be supported by the hidden region if its semantic label is ground (semantic class 1 in our case). Eq. (5.17) – (5.21) enforce the binary variables z to correspond to one of the three cases in Eq. (5.4). To solve this ILP, we make use of Gurobi.

Speeding up inference. While Gurobi is very efficient, it remains too slow for us to handle our typical hierarchies, which contain roughly 200 regions. To address this issue, we therefore propose to first prune the regions. This procedure follows two steps. First, we remove the regions that contain less than 625 pixels, which, based on our statistics, are unlikely to correspond to object instances. Second, we exploit a regressor trained to predict the Intersection over Union (IoU) between a region in the hierarchy and a ground-truth instance. To this end, we make use of a neural network with three fully-connected layers, intertwined with ReLU activation, batch normalization, and dropout. An overiew of deep learning techniques, including the different layers of the types mentioned above, is provided in Section 2.2. This network is depicted by Fig. 5.2. We use deep features in conjunction with hand-crafted geometric ones as input to this shallow IoU regression network. See Section 5.3.3 for more detail about these features. We train this network using the square loss between the true IoU and the predicted one. To this end, we use batches of size 256, a learning rate of 10^{-3} and a momentum of 0.95. The dropout rate was set to 0.5. We also subsample the data so as to have a roughly balanced training set. To this end, we discretize the IoU interval $[0, 1]$ into 10 bins, and subsample the data such that each bin contains roughly the same number of samples. At test time, we keep the 80 regions with highest predicted IoU that satisfy the constraint that each root-to-leaf

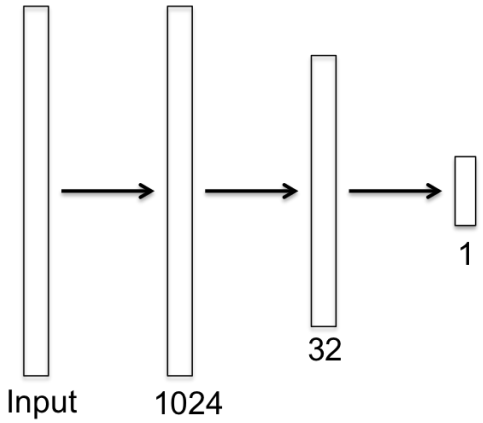


Figure 5.2: **Architecture of our IoU regressor.** We make use of a network with three fully-connected layers to predict the IoU between a candidate region and a ground-truth object instance. We use ReLU activation, batch normalization and dropout after the first and second layers.

path in the segmentation tree contains at least one region. In practice, this pruning yields less than 1% decrease in oracle weighted coverage, while greatly reducing the number of regions.

After pruning, we then train a two-class support classifier on the remaining regions to predict positive or negative support. We make use of this classifier to prune support pairs. To this end, we threshold the classifier score so as to obtain a high recall of positive support. In practice, we achieve 94% recall, while reducing from 5600 to 1100 pairs.

5.3.2 Learning

Given training data, we aim to learn the parameters of our model. One of the challenges of learning comes from the fact that, typically, the ground-truth object instances that we seek to predict do not appear in our hierarchical segmentation. To reflect what will happen at test time, however, we would like to learn our model using the noisy segments from the hierarchies obtained from the training images. To this end, following [Silberman et al., 2014], we rely on an *oracle segmentation*. Below, we first explain how these oracle segmentations are obtained, and then discuss our learning algorithm.

5.3.2.1 Oracle Segmentation

The goal of oracle segmentation is to find among the regions in a noisy hierarchical segmentation those that best match ground-truth object instances and correspond to a valid tree cut, i.e., cover the image without redundancy. To this end, we make use of the ILP formulation of [Silberman et al., 2014]. This formulation relies on two kinds of binary variables. The first ones are equivalent to our region selection variables $a = \{a_{i,l}\}$, $1 \leq i \leq R$, $l \in \{0,1\}$, discussed above. The second kind of variables encode the mapping between ground-truth instances and segments in

the hierarchy. Let us denote these variables as $o \in \mathcal{B}^{G \times R}$, with G the number of ground-truth instances.

An oracle segmentation can then be obtained by solving the optimization problem

$$\underset{a,o}{\text{argmin}} \quad \sum_{g=1}^{G} \sum_{i=1}^{R} \theta^{o}_{g,i} o_{g,i} \tag{5.22}$$

subject to

$$a_{i,l}, o_{g,k} \in \{0,1\}, \quad \forall i, l, g, k, \tag{5.23}$$

$$a_{i,0} + a_{i,1} = 1, \quad \forall i, \tag{5.24}$$

$$\sum_{i \in \gamma} a_{i,1} = 1, \quad \forall \gamma \in \Gamma \tag{5.25}$$

$$o_{g,i} \leq a_{i,1}, \quad \forall g, i, \tag{5.26}$$

$$\sum_{i=1}^{R} o_{g,i} = 1, \quad \forall g, \tag{5.27}$$

$$\begin{aligned} &o_{g,i} + a_{j,1} \leq 1, \quad \forall g, i, j, \\ &\text{if} \quad \text{IoU}(r_g, r_j) > \text{IoU}(r_g, r_i) \end{aligned} \tag{5.28}$$

where $\text{IoU}(\cdot,\cdot)$ denotes the intersection over union between two regions, and $\theta_{g,i} = \frac{|L_{r_g}|}{L}(\text{IoU}(r_g, r_s) - \text{IoU}(r_g, r_i))$ encodes the amount of weighted coverage lost by selecting region i instead of s, which corresponds to the best possible match for ground-truth region g. Most constraints simply force the solution to be valid, with the Eq. (5.28) guaranteeing that, among the regions that are active, the best one is assigned to a ground-truth region.

5.3.2.2 Learning via Structural SVM

We now turn to the learning problem per say. As discussed in Section 2.1.2, structural SVM provides a technique to learn the weights in a CRF. To this end, let $D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(N)}, y^{(N)})\}$ be a set of pairs of images and labels, where $y^{(n)} = \{A^{(n)}, M^{(n)}, S^{(n)}\}$ comprises the best selection of segments from the segmentation tree, obtained using the oracle segmentation described above, the corresponding semantic labels, taken as the dominant label in each region, and support relationships, described in Section 5.4, for image i.

Our goal is to learn the weights in our CRF. The energy in this CRF can be equivalently written as $w^T \phi(x, y)$, where w concatenates all the weights we seek to learn, and, with a slight abuse of notation, $\phi(x, y) = [\phi_a, \phi_{ma}, \phi_s, \phi_{sa}]$ concatenates the corresponding features, so as to compute the different potentials. Following a margin re-scaling structural SVM formulation, learning the weights can be expressed as the

optimization problem

$$\begin{aligned}&\min_{w,\epsilon\leq 0}\frac{1}{2}w^T w+\frac{\lambda}{N}\sum_{n}^{N}\epsilon_n\\&s.t.\ \ w^T[\phi(x^{(n)},y^{(n)})-\phi(x^{(n)},y)]\geq\triangle(y,y^{(n)})-\epsilon_n,\ \forall y\end{aligned}$$

where $\triangle(y,y^{(n)})$ returns the loss of an arbitrary prediction y compared to the best configuration.

Here, to reflect the nature of our problem, where we aim to predict different types of variables jointly, we design the multi-task loss

$$\begin{aligned}\triangle(y,y^{(n)})&=\frac{w_{sup}^{ls}}{Q}\sum_{i=1}^{R}\sum_{j=0}^{R}1[S_{ij}\neq S_{ij}^{*}]\\&+w_r^{ls}\frac{1}{L}\sum_{g\in G}L_{r_g}\left(\max_{i\in A^{(n)}}\text{IoU}(r_g,r_i^{(n)})\right)\\&-w_r^{ls}\frac{1}{L}\sum_{g\in G}L_{r_g}\left(\max_{i\in\hat{A}}\text{IoU}(r_g,r_i)\right)\ ,\end{aligned}\tag{5.29}$$

where $\hat{A}$ is the active set of A, that is, the set of regions such that $a_i=1$, and similarly for $\hat{A}^{(n)}$ w.r.t. $A^{(n)}$. L_{r_g} is the number of pixels in region g, L is the number of pixels in all the ground-truth regions in an image, and Q is the number of active pairs in $\hat{A}$. Here, we use $w_r^{ls}=1, w_{sup}^{ls}=0.5$.

Loss-augmented Inference. An important step in structural SVM learning consists of performing loss-augmented inference to find predictions that have a high loss, but correspond to a low energy (or rather a high score in our maximization formulation). This can be expressed as solving

$$y^{*}=\underset{\hat{y}\in\mathsf{y}}{\text{argmax}}\,\triangle(\hat{y},y^{(n)})+w^T\phi(x,\hat{y})\ .\tag{5.30}$$

Translating this into an ILP then yields the problem

$$\begin{aligned}\underset{a,m,s,o}{\text{argmax}}\quad&\sum_{i=1}^{R}\theta_i^a a_{i,1}+\sum_{i=1}^{R}\sum_{k=0}^{K}\theta_{i,k}^m m_{i,k}+\sum_{i=1}^{R}\sum_{j=0}^{R}\sum_{t=0}^{2}\theta_{i,j,t}^s s_{i,j,t}\\&+\sum_{i=1}^{R}\sum_{j=0}^{R}\sum_{l=1}^{3}\theta_{i,j,k}^{sa}z_{i,j,l}+\sum_{g=1}^{G}\sum_{i=1}^{R}\quad\theta_{g,i}^o o_{g,i}+\sum_{i=1}^{R}\sum_{j=0}^{R}\sum_{t=0}^{2}\theta_{i,j,t}^{sl}s_{i,j,t}\end{aligned}\tag{5.31}$$

subject to the constraints of (5.5) and (5.22). Here, $\theta_{g,i}^o$ encodes the loss on the regions and is defined as in (5.22), $\theta_{i,j,t}^{sl}$ encodes the hamming loss on support relationships.

It can thus be written as

$$\theta^{sl}_{i,j,t} = \frac{1}{Q}, s.t \quad t \neq S^*_{ij} \quad \forall t \in \{1,2,3\} \; . \tag{5.32}$$

To learn our model, we use the BCFW solver of [Lacoste-Julien et al., 2013]. Loss-augmented inference takes 1s per image on average.

5.3.3 Features

As discussed above, the IoU regressor, the support classifier and the potentials of Eq. (5.1) rely on different types of features. Here, we describe these feature vectors.

The IoU regressor relies on four types of features as input, which we refer to as **Conv5-SP**, **Pb-SP**, **Ext-Pb-SP** and **RGeo**. **Conv5-SP** is obtained from spatially pooled [He et al., 2014] features coming from the conv5 layer of the FCN-32s model of [Long et al., 2015] fine-tuned on NYUv2 to predict semantics using RGB and HHA as input. HHAs were obtained from depth prediction using the method of [Eigen et al., 2014]. In particular, the HHA records, for each pixel, the horizontal disparity, height above floor, and the angle of the local surface normal referring to the gravity direction. **Pb-SP** and **Ext-Pb-SP** are derived from the semantic probability maps of the FCN-32s model mentioned above, using spatial pooling on each region and on a bounding box of 1.25 the region's extent around it, respectively. **RGeo** corresponds to the geometry features used in [Silberman et al., 2012a]. For the architecture and parameters of the FCN-32s, please see Section 2.2.2.

The support classifier relies on two types of features. The first concatenates **Pb-SP**, **Ext-Pb-SP** and **RGeo** for both regions. The second, denoted as **PGeo**, includes the containment, geometry and horizontal features of [Silberman et al., 2012a] computed on pairs of regions.

The feature vector f^a_i is obtained by concatenating two types of features, which we refer to as **RF** and **RGeo**. **RF** corresponds to the feature map after the second batch normalization module in the 3-layer neural network described in Section 5.3.1. It encodes the connection between the IoU regressor and the selection of the region.

The feature vector f^{ma}_i contains five types of features, denoted by **RGeo**, **Pb-SP**, **Ext-Pb-SP**, **Pb** and **Hm**. The first three have been described above. **Pb** is defined as the average over the region pixels of the K-dimensional semantic probability vectors obtained by the same FCN-32s as above. **Hm** aims to incorporate dependencies between semantics and support relationships. To this end, for region i, this feature is obtained by averaging over all the other regions j the probability of each support class between i and j, obtained by our SVM support classifier.

The feature vector f^s_{ij} is formed by two feature types, **Ps** and **Pm**. **Ps** is directly taken as the probabilities predicted by our support classifier. **Pm** aims at modeling dependencies between support and semantics. It concatenates the semantic features **Pb** discussed above for both regions.

The feature vector f^{sa}_{ij} concatenates **RGeo** and **RF** features for both regions, as well as the corresponding IoUs predicted by our 3-layer neural network. It further

includes the feature **PGeo** described above.

5.4 Experimental Evaluation

We evaluate our model on the NYUv2 dataset, which provides RGB images and their corresponding depth maps. Note that, here, we do *not* use these depth maps. The dataset contains 749 images for training and 654 for testing.

The ground-truth regions, i.e., object instances, and corresponding semantics are provided by [Silberman et al., 2012a]. The semantics include four classes: ground, structure, props and objects. Ground-truth support relationships were defined by [Silberman et al., 2014] on the ground-truth regions. Based on the strategy of [Silberman et al., 2014], we map these ground-truth support relationships to our segmentation hierarchy [Gupta et al., 2013] as follows: Any pair in which both regions have an IoU with ground-truth regions greater than 0.25 is assigned the corresponding ground-truth type. The other regions are assigned the *no support* label. If, at the end of this procedure, a region is not supported by any other region, we define it as being supported by the hidden one.

The computation cost of our model is affordable. The running time for feature extraction on regions and pairs are 14s and 2.7s per image on average, respectively. Given the features, the pruning process for pairs takes 3s per image and that for regions 0.2s. Inference then takes 0.2s per image.

5.4.1 Evaluation Metrics

Since we predict three different types of variables, we need different metrics to evaluate them. Here, we use:

Instance segmentation accuracy. To evaluate our segmentation results, we make use of the maximum weighted coverage, defined over ground-truth regions $\mathcal{G}$ and predicted regions $\mathcal{R}$ as

$$\text{Coverage}_w(\mathcal{G}, \mathcal{R}) = \frac{1}{|\mathcal{I}|} \sum_{j=1}^{|\mathcal{G}|} |r_j^G| \max_{1,\ldots,|R|} \text{IoU}(r_j^G, r_i^R)$$

where $|\mathcal{I}|$ is the number of pixels in the whole set of ground-truth regions, which may be less than the total number of pixels in the image, and $|r_j^G|$ is number of pixels in ground-truth region j.

Semantic labeling accuracy. To evaluate the predicted semantics, we make use of the standard average accuracy computed over all the pixels and per-class accuracy, where averaging is done over the classes.

Support relationship accuracy. For the support relationships, we evaluate the precision and recall of the positive support types on pairs not containing the hidden

region. These values are defined as

$$\text{precision} = \frac{\text{\# true positive predictions}}{\text{\# positive predictions}}, \tag{5.33}$$

$$\text{recall} = \frac{\text{\# true positive predictions}}{\text{\# of positive samples}}. \tag{5.34}$$

5.4.2 Experimental Results

We now present our results on NYUv2. Since our model addresses multiple tasks, as a first experiment, we evaluate the influence of several of its components via an ablation study. To this end, we compare our complete model (**Ours**) with the following baselines:

Basic: This baseline only performs instance segmentation and includes the region unary and tree constraints of Eq. (5.1).

Ours-NS: This model jointly predicts the region selection variables and the semantics. However, it does not account for the support relationships. This model consists of the first three terms in Eq. (5.1).

Ours-ND: This model also infers the three kinds of variables. It contains all the terms in Eq. (5.1), but does not leverage the features that link support and semantics, i.e., **Hm** and **Ps** in Section 5.3.3. In essence, while predicting all variables, this baseline only models limited dependencies between them. In addition to these baselines, we also report the support predictions obtained with the linear SVM support classifier (**SC**) discussed in Section 5.3.3, which, among others, makes use of features encoding information about the region IoU with ground-truth and the semantics.

The results of our method and of these baselines are provided in Table 5.2. Note that some baselines do not predict all the variables, and can thus not be evaluated according to all the metrics. These results show that (i) jointly predicting regions and semantics improves the quality of the segments; (ii) predicting all three types of variables yields performance improvement to the support quality compared to our support classifier; (iii) modeling the dependencies between the different variable types further improves the support predictions, particularly in terms of recall. Altogether, we believe that these results demonstrate the benefits of jointly inferring regions, semantics and support relationships.

To further evidence the impact of semantics, we performed an experiment where we used the ground-truth ones in our model. This model is denoted as **Ours(GtSem)**. This resulted in a 3.1% relative improvement on recall, thus showing that better semantics yield better support.

In Fig. 5.3, we provide some qualitative results obtained with our approach. Note that the semantic labels we predict closely match the ground-truth ones. Note also that, while they contain some degree of over-segmentation, the regions we produce typically still remain reasonably large, with a clear semantic meaning. Our method is also able to predict accurate support relationships, even in the presence of many different objects, as in the last row of the figure. In Fig. 5.4, we show a typical failure

Model	Oracle W. Cov
[Hoiem et al., 2011]	50.7
[Ren and Shakhnarovich, 2013]	50.7
Ours	64.9

Table 5.1: **Evaluation of the Segmentation Hierarchies Based on NYUv2 RGB:** Here we compare our segmentation hierarchy with several baselines based on monocular image only. We evaluate these hierarchies with the weighted coverage of their corresponding oracle set.

Model	W. Cov	Sem Avg Acc	Sem Per-Cls Acc	Sup. Prec	Sup. Recall
Basic	58.9	-	-	-	-
SC	-	-	-	44.8	39.0
Ours-NS	59.3	73.0	72.0	-	-
Ours-ND	59.3	73.3	72.2	47.0	41.9
Ours	59.4	73.2	72.1	47.6	43.1
Ours(GtSem)	60.1	-	-	48.2	45.0

Table 5.2: **Evaluation on NYUv2.** We compare our approach to several baselines, mostly corresponding to different components of our complete model, where in above table "Sup. Prec" represents support precision, and "Sup. Recall" represents support recall. Note that some of these baselines do not predict all variable types, and can thus only be evaluated on some metrics. These results demonstrate the importance of jointly inferring multiple variable types, in particular on the quality of the support relationships.

case of our approach. We have observed that failures mostly occur when a region is over-segmented, or assigned to the wrong semantic category. Note that this again indicates the dependencies between these different subtasks of indoor scene parsing.

Furthermore, we compare our segmentation hierarchy with baselines that are based on monocular image. In contrast to previous monocular image based segmentation hierarchy [Hoiem et al., 2011; Ren and Shakhnarovich, 2013], we generate the hierarchy utilizing depth predictions [Eigen and Fergus, 2015]. A qualitative comparison is shown in Table 5.1. We evalutate the segmentation hierarchies using the weighted coverage between the ground truth and their corresponding oracle segments. Note that the results of the baselines were taken from [Silberman et al., 2014]. This comparison demonstrates the higher quality of our hierarchy compared to the baselines, and the benefits of utilizing depth prediction for our task.

Comparison with RGBD-based methods. As mentioned previously, existing methods that predict support relationships all work with RGBD images as input. To compare against these methods, we slightly modified our approach to exploit RGBD. In particular, we generated the hierarchy using ground-truth depth, and employed ground-truth depth to extract our features, except for the semantic probability ones. The results in Table 5.4 show again that our model benefits from solving multiple tasks. Note that, despite the fact that the oracle performance obtained from our seg-

Figure 5.3: **Qualitative evaluation of our results.** We show the input image, the ground-truth semantics, the semantics predicted by our approach, and our regions and support predictions. We show the correct relationships in white and the incorrect ones in black. Support from below is indicated by an arrow head and from behind by a diamond one. Note that our semantics match the ground-truth ones quite closely. Furthermore, our regions typically correspond to semantically-meaningful portions of the scene, that is, complete object instances, and our support corresponds to correct relationships. (Best viewed in color.)

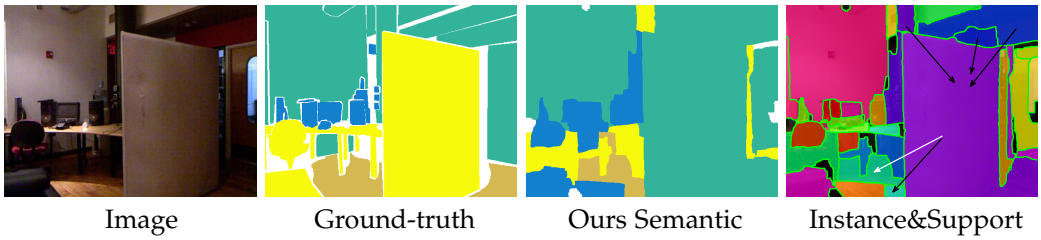


Figure 5.4: **Failure case.** Here, our support relationships are affected by a wrong semantic labeling.

Model	Oracle W.Cov	W. Cov	Sem Avg Acc	Sem Per-Cls Acc
Basic	68.8	61.1	-	-
Ours-NS	68.8	62.8	74.8	73.7
Ours	68.8	62.7	75.3	74.3
[Silberman et al., 2014]	70.6	62.5	-	-

Table 5.3: **Evaluation on NYUv2 RGBD.** We compare our approach to several baselines corresponding to different components of our complete model and to the state-of-the-art methods [Silberman et al., 2014]. Note that, while our oracle weighted coverage is lower than that of [Silberman et al., 2014], we achieve higher weighted coverage, thus showing the impact of accounting for the dependencies between multiple tasks.

Model	Support Precision	Support Recall
SC	48.3	37.9
Ours	49.5	38.6
[Silberman et al., 2012a]	54.5	-

Table 5.4: **Support Evaluation on NYUv2 RGBD.** We compare our approach to the baseline and state-of-the-art corresponding to [Silberman et al., 2012a]. Note that, Ours evaluates the support relationship on segmentation hierarchy, it is not fair engouth to compare with [Silberman et al., 2012a] which worked on a different fixed segmentation.

mentation hierarchy is lower than that of [Silberman et al., 2014], the segmentation obtained by our method has a higher weighted coverage. In other words, since the gap between our weighted coverage and the oracle one is significantly smaller than for [Silberman et al., 2014], i.e., 6.1% vs 8.1%, our model essentially selects better regions than [Silberman et al., 2014]. The comparison to [Silberman et al., 2012a] for support prediction should be taken with caution, since the regions are different. We believe that this comparison shows that both method perform similarly, with our approach providing additional information about the scene. Note that we expect that exploiting depth more thoroughly than done here could give our approach a bigger boost.

5.5 Conclusion

In this chapter, we have introduced an approach to jointly segmenting the object instances in an image and predicting their semantic labels and support relationships. To the best of our knowledge, this constitutes the first attempt at jointly tackling these three subtasks of the indoor scene understanding problem. Our experiments have demonstrated that jointly reasoning about these three tasks is in general beneficial, and particularly so for support relationships. In addition, we also have evidenced the advantage of utilizing depth prediction for our task. Indoor scene understand-

ing, however, is not limited to these three tasks. One can, for example, also aim at predicting depth, surface normals and object affordances. Ultimately, we believe that all these problems should be tackled jointly to better leverage their dependencies.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The overall goal of this thesis was to understand an indoor scene from a single monocular image. To this end, we have investigated methods that extract useful information in 2D and 3D. In Chapter 3, we tackled the task of depth estimation from a monocular image. Predicted depth enables effective scene understanding in 3D and high quality predictions on related 2D tasks, such as instance segmentation. Based on depth estimates, we generated 3D object box proposals in Chapter 4, and tackled the scene parsing task with instance segmentation, semantic labeling and support relationship inference in 2D in Chapter 5. We summarize the contributions of each work below.

In Chapter 3, we introduced a novel model that estimates detailed depth maps by leveraging high-level scene structures. In particular, we introduced a novel hierarchical representation of the scene which models the scene structure at local, mid-level, and global scales. Compared to earlier works that reason locally, our approach achieves dramatically better performance.

In Chapter 4, we designed the first monocular image 3D box proposal network for indoor scenes with a novel fully differentiable framework, at the core of whose lies a newly-designed residual, differentiable volumetric representation network.

In Chapter 5, we proposed a novel integrated model acting on segmentation hierarchy for jointly reasoning about three subtasks of scene parsing, i.e., instance segmentation, semantic prediction of instances, and support relationship inference on the hierarchy. Here, dependencies of the three subtasks were well exploited. In this framework, we achieved better instance segmentation and more accurate support relationship predictions than methods that tackles these subtasks independently.

Overall, contributions of this thesis are: 1) efforts to overcome the difficulties arising from the ambiguity of depth estimation from a monocular image; 2) exploiting the dependencies on predicted depth of other scene understanding tasks, such as 3D box proposal generation and instance segmentation; 3) novel methods that beat existing ones on several 2D and 3D tasks; 4) rich and effective scene representations at various scales and perspectives.

6.2 Future Work

For comprehensive scene understanding, many tasks remain to be addressed. Below, we discuss several research directions related to the problem tackled in this thesis.

For depth estimation, in this thesis, we proposed a depth estimation method based on traditional hand-crafted features and a traditional inference technique. In the framework of our model, potential improvements can be achieved by exploiting better CRF potentials. In recent years, however, hand-crafted features have been replaced by features learned using deep convolutional networks. In this context, a potential direction for depth estimation could be to exploit other cues that can faciliate this task, such as CAD models and semantics, and incorporate the additonal cues in the deep network.

Our 3D box proposal method from a monocular image, it surfers from inaccuracies due to imperfect depth estimation, both in global and local range. As a consequence, the direction discussed above for depth estimation may improve monocular-image 3D box proposal generation. Furthermore, we observed that, in our experiments, the intermediate depth output of our integrated framework mainly focuses on foreground objects, so as to generate better 3D object proposals. This motivates us to design a depth estimation loss that pays closer attention to foreground. Since our 3D box proposal generation also suffers from the inaccuracy of the scene orientation estimate, which we used as orientation for each individual object, the development of better orientation estimation methods for individual objects can lead to more accurate 3D box proposals. Furthermore, going beyond our framework, we believe that combinations of 2D and 3D features may lead to better performance. To this end, motivated by the spatial transformation network of [Jaderberg et al., 2015], we could learn the model in world coordinates, so as to extract more consistent features for object of the same category and generate 3D box proposals of better quality. This network could further be extended for doing proposal generation and object detection.

For our task of scene parsing, we exploited three related scene parsing subtasks in an intergrated graphical model. A potential avenue for future research would be to incorporate other related tasks. In particular, we trained the features for different potentials and the structural SVM in two independent stages. It would therefore be interesting to investigate the effect of joint learning for depth estimation, feature extraction, and prediction on the three subtasks.

Bibliography

2014. RMRC challenge 2014. http://cs.nyu.edu/~silberman/rmrc2014/. (cited on pages 30 and 36)

Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; and Suesstrunk, S., 2012. Slic superpixels compared to state-of-the-art superpixel methods. *TPAMI*, (2012). (cited on pages 31 and 63)

Arbelaez, P.; Maire, M.; Fowlkes, C.; and Malik, J., 2011. Contour detection and hierarchical image segmentation. *TPAMI*, 33, 5 (2011), 898–916. (cited on pages 29, 63, and 64)

Arbeláez, P.; Pont-Tuset, J.; Barron, J. T.; Marques, F.; and Malik, J., 2014. Multiscale combinatorial grouping. In *CVPR*, 328–335. (cited on pages 5, 62, 63, and 64)

Baig, M. H.; Jagadeesh, V.; Piramuthu, R.; Bhardwaj, A.; Di, W.; and Sundaresan, N., 2014. Im2depth: Scalable exemplar based depth transfer. In *WACV*, 145–152. IEEE. (cited on page 22)

Baig, M. H. and Torresani, L., 2016. Coupled depth learning. In *WACV*, 1–10. IEEE. (cited on page 36)

Bansal, A.; Russell, B.; and Gupta, A., 2016. Marr revisited: 2d-3d alignment via surface normal prediction. *CVPR*, (2016). (cited on pages 1 and 43)

Bishop, C. M., 2006. *Pattern recognition and machine learning*. springer. (cited on pages 9 and 25)

Bosch, A.; Zisserman, A.; and Munoz, X., 2007. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, 401–408. ACM. (cited on page 25)

Bottou, L., 2012. *Stochastic Gradient Descent Tricks*, vol. 7700, 430âĂŞ445. Springer. https://www.microsoft.com/en-us/research/publication/stochastic-gradient-tricks/. (cited on page 13)

Boykov, Y.; Veksler, O.; and Zabih, R., 2001. Fast approximate energy minimization via graph cuts. *TPAMI*, 23, 11 (2001), 1222–1239. (cited on page 9)

Chen, L.-C.; Schwing, A.; Yuille, A.; and Urtasun, R., 2015. Learning deep structured models. In *International Conference on Machine Learning*, 1785–1794. (cited on page 64)

Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Fidler, S.; and Urtasun, R., 2016. Monocular 3d object detection for autonomous driving. In *CVPR*. (cited on pages 41 and 42)

Chen, X.; Kundu, K.; Zhu, Y.; Ma, H.; Fidler, S.; and Urtasun, R., 2017. 3d object proposals using stereo imagery for accurate object class detection. *TPAMI*, (2017). (cited on pages 41 and 42)

Cheng, M.-M.; Zhang, Z.; Lin, W.-Y.; and Torr, P., 2014. Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*. (cited on page 42)

Collins, M.; Schapire, R. E.; and Singer, Y., 2002. Logistic regression, adaboost and bregman distances. *Machine Learning*, 48, 1 (2002), 253–285. (cited on page 26)

Comaniciu, D. and Meer, P., 2002. Mean shift: A robust approach toward feature space analysis. *TPAMI*, 24, 5 (2002), 603–619. (cited on page 63)

Coughlan, J. M. and Yuille, A. L., 1999. Manhattan world: Compass direction from a single image by bayesian inference. In *ICCV*, vol. 2, 941–947. IEEE. (cited on pages 1 and 23)

Cutting, J. E. and Vishton, P. M., 1995. Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth. In *Perception of space and motion*, 69–117. Elsevier. (cited on page 19)

Dai, J.; He, K.; and Sun, J., 2016. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*. (cited on pages 5, 42, and 64)

Dasgupta, S.; Fang, K.; Chen, K.; and Savarese, S., 2016. Delay: Robust spatial layout estimation for cluttered indoor scenes. In *CVPR*, 616–624. (cited on page 1)

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*, 248–255. IEEE. (cited on page 12)

Deng, Z. and Latecki, L. J., 2017. Amodal detection of 3d objects: Inferring 3d bounding boxes from 2d ones in rgb-depth images. In *CVPR*. (cited on page 42)

Duchi, J.; Hazan, E.; and Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, Jul (2011), 2121–2159. (cited on page 13)

Dwibedi, D.; Malisiewicz, T.; Badrinarayanan, V.; and Rabinovich, A., 2016. Deep cuboid detection: Beyond 2d bounding boxes. *arXiv preprint arXiv:1611.10010*, (2016). (cited on page 40)

Eigen, D. and Fergus, R., 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*. (cited on pages xiv, xv, 5, 6, 16, 17, 39, 41, 42, 43, 44, 50, 56, 62, 64, and 75)

Eigen, D.; Puhrsch, C.; and Fergus, R., 2014. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2366–2374. (cited on pages 1, 5, 20, 21, 31, 34, 36, 42, 43, 48, 50, 53, and 72)

Fang, Y.; Xie, J.; Dai, G.; Wang, M.; Zhu, F.; Xu, T.; and Wong, E., 2015. 3d deep shape descriptor. In *CVPR*, 2319–2328. (cited on page 42)

Fidler, S.; Dickinson, S.; and Urtasun, R., 2012. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *NIPS*. (cited on pages 41 and 42)

Fouhey, D. F.; Gupta, A.; and Hebert, M., 2013. Data-driven 3d primitives for single image understanding. In *ICCV*, 3392–3399. IEEE. (cited on pages 23 and 31)

Fouhey, D. F.; Gupta, A.; and Hebert, M., 2014. Unfolding an indoor origami world. In *ECCV*, 687–702. Springer. (cited on pages 21 and 23)

Girdhar, R.; Fouhey, D. F.; Rodriguez, M.; and Gupta, A., 2016. Learning a predictable and generative vector representation for objects. In *ECCV*. (cited on page 42)

Girshick, R., 2015. Fast r-cnn. In *ICCV*. (cited on pages 5, 42, and 48)

Guo, R. and Hoiem, D., 2013. Support surface prediction in indoor scenes. In *ICCV*, 2144–2151. (cited on pages 1, 5, 62, and 64)

Gupta, A.; Efros, A. A.; and Hebert, M., 2010a. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 482–496. Springer. (cited on pages 1 and 21)

Gupta, A.; Hebert, M.; Kanade, T.; and Blei, D. M., 2010b. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 1288–1296. (cited on page 21)

Gupta, S.; Arbeláez, P.; Girshick, R.; and Malik, J., 2015. Aligning 3d models to rgb-d images of cluttered scenes. In *CVPR*. (cited on pages 2 and 42)

Gupta, S.; Arbelaez, P.; and Malik, J., 2013. Perceptual organization and recognition of indoor scenes from rgb-d images. In *CVPR*, 564–571. (cited on pages 5, 62, 64, 65, and 73)

Gupta, S.; Girshick, R.; Arbeláez, P.; and Malik, J., 2014a. Learning rich features from rgb-d images for object detection and segmentation. In *ECCV*. (cited on pages 2 and 51)

Gupta, S.; Girshick, R.; Arbelaez, P.; and Malik, J., 2014b. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*. (cited on page 29)

Hayder, Z.; He, X.; and Salzmann, M., 2016. Learning to co-generate object proposals with a deep structured network. In *CVPR*. (cited on page 42)

Hayder, Z.; He, X.; and Salzmann, M., 2017. Boundary-aware instance segmentation. In *CVPR*. (cited on page 42)

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R., 2017. Mask r-cnn. *ICCV*, (2017). (cited on pages 5 and 42)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2014. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 346–361. Springer. (cited on page 72)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2016. Deep residual learning for image recognition. In *CVPR*, 770–778. (cited on pages 12 and 14)

He, X.; Zemel, R. S.; and Carreira-Perpiñán, M. Á., 2004. Multiscale conditional random fields for image labeling. In *CVPR*, vol. 2, II–II. IEEE. (cited on page 7)

Hedau, V.; Hoiem, D.; and Forsyth, D., 2009. Recovering the spatial layout of cluttered rooms. In *ICCV*, 1849–1856. IEEE. (cited on pages 1, 23, and 29)

Hedau, V.; Hoiem, D.; and Forsyth, D., 2010. Thinking inside the box: Using appearance models and context based on room geometry. *ECCV*, (2010). (cited on pages 1, 21, 23, and 40)

Hoiem, D.; Efros, A. A.; and Hebert, M., 2005. Geometric context from a single image. In *ICCV*, vol. 1, 654–661. IEEE. (cited on pages 21 and 23)

Hoiem, D.; Efros, A. A.; and Hebert, M., 2007a. Recovering surface layout from an image. *IJCV*, 75, 1 (2007), 151–172. (cited on pages 1, 21, and 23)

Hoiem, D.; Efros, A. A.; and Hebert, M., 2011. Recovering occlusion boundaries from an image. *IJCV*, 91, 3 (2011), 328–346. (cited on pages 6, 64, and 75)

Hoiem, D.; Stein, A. N.; Efros, A. A.; and Hebert, M., 2007b. Recovering occlusion boundaries from a single image. In *ICCV*, 1–8. IEEE. (cited on page 26)

Hu, S. X.; Williams, C. K.; and Todorovic, S., 2015. Tree-cut for probabilistic image segmentation. *arXiv preprint arXiv:1506.03852*, (2015). (cited on page 64)

Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q., 2017. Densely connected convolutional networks. In *CVPR*. (cited on page 12)

Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456. (cited on page 14)

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al., 2015. Spatial transformer networks. In *NIPS*, 2017–2025. (cited on page 80)

Jia, Z.; Gallagher, A.; Saxena, A.; and Chen, T., 2013. 3d-based reasoning with blocks, support, and stability. In *CVPR*, 1–8. (cited on pages 5, 62, and 64)

Karsch, K.; Liu, C.; and Kang, S. B., 2012. Depth extraction from video using non-parametric sampling. In *ECCV*. (cited on pages 4, 7, 20, 21, 22, 30, and 42)

Kingma, D. and Ba, J., 2014. Adam: A method for stochastic optimization. *ICLR*, (2014). (cited on page 13)

Koenderink, J. J., 1998. Pictorial relief. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 356, 1740 (1998), 1071–1086. (cited on page 19)

Koenderink, J. J.; Van Doorn, A. J.; and Kappers, A. M., 1996. Pictorial surface attitude and local depth comparisons. *Perception & Psychophysics*, 58, 2 (1996), 163–173. (cited on page 19)

Koller, D. and Friedman, N., 2009. Probabilistic graphical models: Principles andtechniques. In *MIT press*. (cited on page 8)

Koltun, V., 2011. Efficient inference in fully connected crfs with gaussian edge potentials. *NIPS*, (2011). (cited on page 64)

Konrad, J.; G.Brown; Wang, M.; Ishwar, P.; Wu, C.; and Mukherjee, D., 2012a. Automatic 2d-to-3d image conversion using 3d examples from the internet. In *SPIE Stereoscopic Displays and Applications*. (cited on page 22)

Konrad, J.; Wang, M.; and Ishwar, P., 2012b. 2d-to-3d image conversion by learning depth from examples. In *3DCINE*. (cited on page 22)

Krähenbühl, P. and Koltun, V., 2011. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*, 109–117. (cited on page 9)

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, 1097–1105. (cited on pages 11, 12, and 16)

Lacoste-Julien, S.; Jaggi, M.; Schmidt, M.; and Pletscher, P., 2013. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*. (cited on pages 11 and 72)

Ladicky, L.; Shi, J.; and Pollefeys, M., 2014. Pulling things out of perspective. In *CVPR*. (cited on pages 4, 20, 21, 22, 30, 31, and 42)

Ladickỳ, L.; Zeisl, B.; and Pollefeys, M., 2014. Discriminatively trained dense surface normal estimation. In *ECCV*, 468–484. Springer. (cited on pages 23 and 64)

Lafferty, J.; McCallum, A.; and Pereira, F. C., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001). (cited on page 7)

Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; and Navab, N., 2016. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 239–248. IEEE. (cited on pages 21, 42, and 43)

LeCun, Y.; Boser, B. E.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W. E.; and Jackel, L. D., 1990. Handwritten digit recognition with a back-propagation network. In *NIPS*, 396–404. (cited on page 11)

Lee, D. C.; Gupta, A.; Hebert, M.; and Kanade, T., 2010. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 1288–1296. (cited on page 23)

Lee, D. C.; Hebert, M.; and Kanade, T., 2009. Geometric reasoning for single image structure recovery. In *CVPR*, 2136–2143. IEEE. (cited on pages 1, 21, 23, and 29)

Li, B.; Shen, C.; Dai, Y.; van den Hengel, A.; and He, M., 2015. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *CVPR*, 1119–1127. (cited on pages 21 and 43)

Li, L.-J.; Su, H.; Fei-Fei, L.; and Xing, E. P., 2010. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *NIPS*, 1378–1386. (cited on page 25)

Lin, D.; Fidler, S.; and Urtasun, R., 2013. Holistic scene understanding for 3d object detection with rgbd cameras. In *ICCV*. (cited on pages 2 and 42)

Lin, G.; Shen, C.; van den Hengel, A.; and Reid, I., 2016. Efficient piecewise training of deep structured models for semantic segmentation. In *CVPR*, 3194–3203. (cited on page 64)

Liu, B.; Gould, S.; and Koller, D., 2010. Single image depth estimation from predicted semantic labels. In *CVPR*, 1253–1260. IEEE. (cited on pages 4, 20, 21, and 22)

Liu, F.; Shen, C.; and Lin, G., 2015. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, 5162–5170. (cited on pages 21 and 43)

Liu, F.; Shen, C.; Lin, G.; and Reid, I., 2016. Learning depth from single monocular images using deep convolutional neural fields. *TPAMI*, 38, 10 (2016). (cited on page 42)

Liu, M.; Salzmann, M.; and He, X., 2014. Discrete-continuous depth estimation from a single image. In *CVPR*. (cited on pages 4, 7, 20, 21, 22, 25, 27, 30, and 42)

Long, J.; Shelhamer, E.; and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In *CVPR*, 3431–3440. (cited on pages 2, 5, 12, 16, 17, 62, 64, and 72)

Mallya, A. and Lazebnik, S., 2015. Learning informative edge maps for indoor scene layout prediction. In *ICCV*, 936–944. (cited on page 1)

Maturana, D. and Scherer, S., 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 922–928. IEEE. (cited on page 42)

Nedovic, V.; Smeulders, A. W.; Redert, A.; and Geusebroek, J.-M., 2010. Stages as models of scene geometry. *TPAMI*, 32, 9 (2010), 1673–1687. (cited on page 23)

Nemhauser, G. L. and Wolsey, L. A., 1988. Integer programming and combinatorial optimization. *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, 20 (1988), 8–12. (cited on page 10)

Nesterov, Y., 1983. A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2). In *Doklady AN USSR*, vol. 269, 543–547. (cited on page 13)

Newcombe, R. A.; Izadi, S.; Hilliges, O.; Molyneaux, D.; Kim, D.; Davison, A. J.; Kohi, P.; Shotton, J.; Hodges, S.; and Fitzgibbon, A., 2011. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*. (cited on page 44)

Ohta, Y., 1985. *Knowledge-based interpretation of outdoor natural color scenes*, vol. 4. Morgan Kaufmann. (cited on page 20)

Oliva, A. and Torralba, A., 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42, 3 (2001), 145–175. (cited on page 25)

Payet, N. and Todorovic, S., 2011. From contours to 3d object detection and pose estimation. In *ICCV*. (cited on page 40)

Pinheiro, P. O.; Collobert, R.; and Dollár, P., 2015. Learning to segment object candidates. In *NIPS*. (cited on page 42)

Pont-Tuset, J.; Arbelaez, P.; Barron, J. T.; Marques, F.; and Malik, J., 2017. Multi-scale combinatorial grouping for image segmentation and object proposal generation. *TPAMI*, 39, 1 (2017). (cited on page 42)

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. (2017). (cited on page 42)

Qi, C. R.; Su, H.; Niessner, M.; Dai, A.; Yan, M.; and Guibas, L. J., 2016. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 5648–5656. (cited on page 42)

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, (2017). (cited on page 42)

Qian, N., 1999. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12, 1 (1999), 145–151. (cited on page 13)

Ren, M. and Zemel, R. S., 2017. End-to-end instance segmentation with recurrent attention. (2017). (cited on page 64)

Ren, S.; He, K.; Girshick, R.; and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*. (cited on pages 5, 40, 42, 47, 50, and 51)

Ren, X.; Bo, L.; and Fox, D., 2012. Rgb-(d) scene labeling: Features and algorithms. In *CVPR*, 2759–2766. IEEE. (cited on pages 29 and 64)

Ren, Z. and Shakhnarovich, G., 2013. Image segmentation by cascaded region agglomeration. In *CVPR*, 2011–2018. (cited on pages 6 and 75)

Ren, Z. and Sudderth, E. B., 2016. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *CVPR*, 1525–1533. (cited on pages 1 and 48)

Roberts, L. G., 1963. *Machine perception of three-dimensional solids*. Ph.D. thesis, Massachusetts Institute of Technology. (cited on page 23)

Roth, D. and Yih, W.-t., 2005. Integer linear programming inference for conditional random fields. In *ICML*, 736–743. ACM. (cited on page 9)

Rother, C., 2002. A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20, 9 (2002), 647–655. (cited on page 24)

Roy, A. and Todorovic, S., 2016. Monocular depth estimation using neural regression forest. In *CVPR*, 5506–5514. (cited on pages 21 and 43)

Saxena, A.; Chung, S. H.; and Ng, A. Y., 2007. 3-d depth reconstruction from a single still image. *IJCV*, (2007). (cited on pages 4, 20, 21, and 22)

Saxena, A.; Sun, M.; and Ng, A. Y., 2009. Make3d: Learning 3d scene structure from a single still image. *TPAMI*, (2009). (cited on pages 4, 20, and 22)

Schwing, A.; Hazan, T.; Pollefeys, M.; and Urtasun, R., 2011. Distributed message passing for large scale graphical models. In *CVPR*. (cited on pages 10 and 30)

Schwing, A. G.; Fidler, S.; Pollefeys, M.; and Urtasun, R., 2013. Box in the box: Joint 3d layout and object reasoning from single images. In *ICCV*. (cited on pages 1, 21, and 23)

Shi, J. and Malik, J., 2000. Normalized cuts and image segmentation. *TPAMI*, 22, 8 (2000), 888–905. (cited on pages 5, 62, 63, and 64)

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R., 2012a. Indoor segmentation and support inference from rgbd images. In *ECCV*, 746–760. Springer. (cited on pages xviii, 1, 2, 5, 7, 32, 50, 62, 63, 64, 65, 72, 73, and 77)

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R., 2012b. Indoor segmentation and support inference from rgbd images. In *ECCV*. (cited on page 30)

Silberman, N.; Sontag, D.; and Fergus, R., 2014. Instance segmentation of indoor scenes using a coverage loss. In *ECCV*, 616–631. Springer. (cited on pages xviii, 7, 62, 64, 65, 69, 73, 75, and 77)

Simard, P. Y.; Steinkraus, D.; Platt, J. C.; et al., 2003. Best practices for convolutional neural networks applied to visual document analysis. In *ICDAR*, vol. 3, 958–962. (cited on page 11)

Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, (2014). (cited on pages xiv, 12, 15, and 16)

Song, S.; Lichtenberg, S. P.; and Xiao, J., 2015. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*. (cited on pages 2, 50, and 56)

Song, S. and Xiao, J., 2016. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*. (cited on pages xiv, 1, 2, 15, 16, 17, 41, 42, 44, 47, 50, and 55)

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A., 2015. Going deeper with convolutions. In *CVPR*, 1–9. (cited on pages 12, 14, and 16)

Tighe, J. and Lazebnik, S., 2010. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*, 352–365. Springer. (cited on pages 28 and 64)

Torralba, A. and Oliva, A., 2002. Depth estimation from image structure. *TPAMI*, 24, 9 (2002), 1226–1238. (cited on page 23)

Tsochantaridis, I.; Joachims, T.; Hofmann, T.; and Altun, Y., 2005. Large margin methods for structured and interdependent output variables. *Journal of machine learning research*, 6, Sep (2005), 1453–1484. (cited on page 11)

Uijlings, J. R.; van de Sande, K. E.; Gevers, T.; and Smeulders, A. W., 2013. Selective search for object recognition. *IJCV*, 104, 2 (2013). (cited on pages 42, 49, and 55)

Vedaldi, A. and Lenc, K., 2015. Matconvnet: Convolutional neural networks for matlab. In *ACM*, 689–692. ACM. (cited on pages xiii, 12, and 13)

Wang, P.; Shen, X.; Lin, Z.; Cohen, S.; Price, B.; and Yuille, A. L., 2015a. Towards unified depth and semantic prediction from a single image. In *CVPR*, 2800–2809. (cited on pages 21 and 43)

Wang, X.; Fouhey, D.; and Gupta, A., 2015b. Designing deep networks for surface normal estimation. In *CVPR*, 539–547. (cited on page 43)

Weiss, Y.; Yanover, C.; and Meltzer, T., 2007. Map estimation, linear programming and belief propagation with convex free energies. *Proc. UAI*, (2007). (cited on page 9)

Wu, J.; Xue, T.; Lim, J. J.; Tian, Y.; Tenenbaum, J. B.; Torralba, A.; and Freeman, W. T., 2016. Single image 3d interpreter network. In *ECCV*. (cited on page 42)

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J., 2015. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 1912–1920. (cited on page 42)

Xu, D.; Ricci, E.; Ouyang, W.; Wang, X.; and Sebe, N., 2017. Multi-scale continuous crfs as sequential deep networks for monocular depth estimation. *CVPR*, (2017). (cited on pages 21 and 43)

Yang, M. Y.; Liao, W.; Ackermann, H.; and Rosenhahn, B., 2017. On support relations and semantic scene graphs. *ISPRS Journal of Photogrammetry and Remote Sensing*, 131 (2017), 15–25. (cited on pages 5, 62, and 64)

Zeiler, M. D., 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, (2012). (cited on page 13)

Zhao, H.; Shi, J.; Qi, X.; Wang, X.; and Jia, J., 2017. Pyramid scene parsing network. (2017). (cited on page 64)

Zhao, Y. and Zhu, S.-C., 2013. Scene parsing by integrating function, geometry and appearance models. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 3119–3126. IEEE. (cited on page 1)

Zheng, S.; Jayasumana, S.; Romera-Paredes, B.; Vineet, V.; Su, Z.; Du, D.; Huang, C.; and Torr, P. H., 2015. Conditional random fields as recurrent neural networks. In *ICCV*, 1529–1537. (cited on pages 5, 62, and 64)

Zhou, T.; Brown, M.; Snavely, N.; and Lowe, D. G., 2017. Unsupervised learning of depth and ego-motion from video. In *CVPR*. (cited on page 42)

Zhuo, W.; Salzmann, M.; He, X.; and Liu, M., 2015. Indoor scene structure analysis for single image depth estimation. In *CVPR*. (cited on pages 1, 3, 10, 22, and 42)

Zhuo, W.; Salzmann, M.; He, X.; and Liu, M., 2017. Indoor scene parsing with instance segmentation, semantic labeling and support relationship inference. In *CVPR*. (cited on pages 4, 5, and 9)

Zimmerman, G. L.; Legge, G. E.; and Cavanagh, P., 1995. Pictorial depth cues: A new slant. *JOSA A*, 12, 1 (1995), 17–26. (cited on page 19)