Holistic Indoor Scene Understanding By Context Supported Instance Segmentation

By

Lin Guo

Bachelor of Science in Electrical Engineering
Tianjin University
Tianjin, China
2012

Master of Science in Electrical Engineering
Oklahoma State University
Stillwater, Oklahoma, USA
2015

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2020

Holistic Indoor Scene Understanding By Context Supported Instance Segmentation

Dissertation Approved:

Dr. Guoliang Fan

Dissertation Advisor

Dr. Martin Hagan

Dr. Weihua Sheng

Dr. Jamey Jacob

ACKNOWLEDGMENTS

I would like to express my sincere thanks to my advisor, Dr. Guoliang Fan, for his patient guidance, enthusiasm for research and sharing valuable experience. His thoughtful advice on the research have had more of an impact in my life. Also I would like to thank my fellow graduate students, Liangjiang, Mahdi, Nate, Jing and Le for helping my research work and being the best co-workers. My sincere thanks must also go to my PhD advisory committee, Dr. Martin Hagan, Dr. Weihua Sheng, and Dr. Jamey D. Jacob for their dedication and commitment throughout the review of this dissertation. I greatly appreciate their extremely helpful guidance during the past few years.

Acknowledgements reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

Name: Lin Guo

Date of Degree: December, 2020

Title of Study: Holistic Indoor Scene Understanding By Context Supported Instance Segmentation

Major Field: ELECTRICAL ENGINEERING

Abstract: Intelligent robots require advanced vision capabilities to perceive and interact with the real physical world. While computer vision has made great strides in recent years, its predominant paradigm still focuses on building deep-learning networks or handcrafted features to achieve semantic labeling or instance segmentation separately and independently. However, the two tasks should be synergistically unified in the recognition flow since they have a complementary nature in scene understanding.

This dissertation presents the detection of instances in multiple scene understanding levels. Representations that enable intelligent systems to not only recognize what is seen (e.g. Does that pixel represent a chair?), but also predict contextual information about the complete 3D scene as a whole (e.g. How big is the chair? Is the chair placed next to a table?). More specifically, it presents a flow of understanding from local information to global fitness. First, we investigate in the 3D geometry information of instances. A new approach of generating tight cuboids for objects is presented. Then, we take advantage of the trained semantic labeling networks by using the intermediate layer output as a per-category local detector. Instance hypotheses are generated to help traditional optimization methods to get a higher instance segmentation accuracy. After that, to bring the local detection results to holistic scene understanding, our method optimizes object instance segmentation considering both the spacial fitness and the relational compatibility. The context information is implemented using graphical models which represent the scene level object placement in three ways: horizontal, vertical and non-placement hanging relations. Finally, the context information is implemented to a network structure. A deep learning-based reinferencing frame work is proposed to boost any pixel-level labeling outputs using our local collaborative object presence (LoCOP) feature as the global-to-local guidance.

This dissertation demonstrates that uniting pixel-level detection and instance segmentation not only significantly improves the overall performance for localized and individualized analysis, but also paves the way for holistic scene understanding.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

Table Page

Table Page

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

1.1 Motivation and background

Computer vision is a powerful tool in robotics in recent years [13–16]. Much research has been conducted to make robots to be co-inhabitants or co-workers. Previously, assistance robots wait for human-issued commands, and the human-robot interaction (HRI) can only be done in a close way. Under this circumstance, assistance robots have a limited sense of the environment that they are in. The robot is a human-like co-inhabitant, serves as a housekeeper, and it can do its work without human's consecutive commands. On contrast, the traditional robot, which follows the owner's commands from time to time, makes people feel being monitored without much privacy. In order to understand and comprehensive tasks, the robots need to have good understanding about the scene they are in [1, 17–19].

Recently, a number of researchers focus on developing a 3D map with the help of fast development of depth sensors. However, no object information is provided by the reconstructions. The robot needs a semantic map, which provides a good understanding of the house. Thus, it can arrange the work for itself. For example, after finding the owner is making breakfast in the kitchen, the robot moves to the bedroom to do some organization. So the scene understanding plays an important role for the robot to assistant people's living.

Imagine a domestic robot preparing to set a dining table. Which piece of visual information would it find to be more useful for the task? Seeing a table on the left and chairs on the right, or seeing a table two meters away from me behind three chairs, the

Figure 1.1: Robot for assisted living example [4].

tabletop is one meter above the door, and there is enough empty space on the table to place the dishes. While performing complex tasks such as preparing dining tables, autonomous robotic systems would typically benefit from a complete 3D visual understanding holistically in the scene: their locations and orientations, accessible space, and spaces acquired by all the objects in the scene. However, most computer vision algorithms will only produce information to the extent of local relations like *table on the left and chairs on the right* from 2D images.This highlights a fundamental limitation behind classic 2D image-centric computer vision tasks: they are targeted at understanding 2D images, but not the 3D physical world behind them. Moreover, since images are only 2D partial representations of complete 3D scenes, they can exhibit dramatic variations from minor changes to camera viewpoint, materials, lighting, and object arrangements, which continue to obscure image recognition algorithms. In our research, we expect to acquire the relation between indoor objects which provides holistic scene understanding, and then locate the objects in the 3D space.

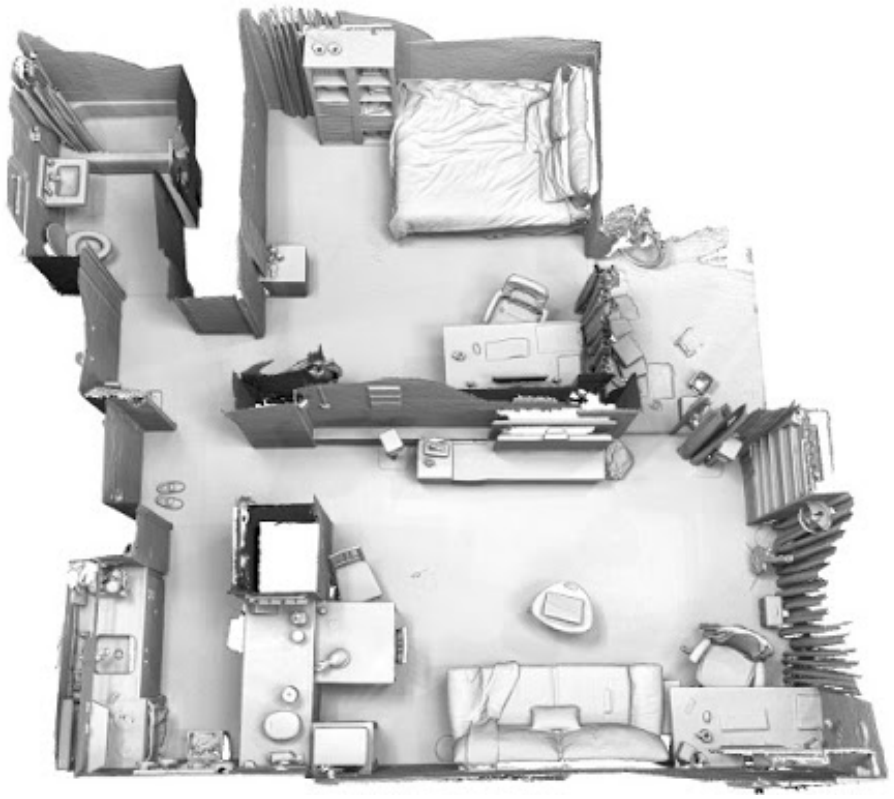

Figure 1.2: An indoor 3D reconstruction example [5]. The reconstruction provides an accurate scene layout but with no object label information.

1.2 Research objectives and approach

My research is uniquely defined by the following aspects:

From 2D images to 3D models: Explore the direct use of 3D data as both input and output presentations for computer vision algorithms, instead of reasoning over 2D image pixels, where the information is limited by the field of view.

From low-level pattern to high-level meaning: Introduce measures and models with strong real-life meaning. Incorporate human's understanding to the scene and ensure the extrapolation availability at the same time.

From local to holism: Make use of contextual information beyond single objects. Understand the object pair patterns from small range up to big range, which is the level of scenes.

Figure 1.3: An indoor 2D semantic map eample [6]. The indoor objects are marked using different color.

The core problem of indoor scene understanding is about knowing what objects are in the scene, where they are, and why they are set in such ways. The goal of this dissertation is to develop computer vision algorithms that can understand the visual world in terms of both low-level 3D structure and high-level semantics of indoor instances. More importantly, the system should not only be able to recognize what it sees, but also be able to reason contextual information related to its complete 3D environment - including regions beyond the visible surfaces in the view, such as: what to expect in the given scene. Towards this goal, this thesis aims to develop 3D instance level representations and relations from RGB-Depth data captured from 3D depth sensors such as the Microsoft Kinect, and it is our goal to provide useful 3D understanding for real-world applications.

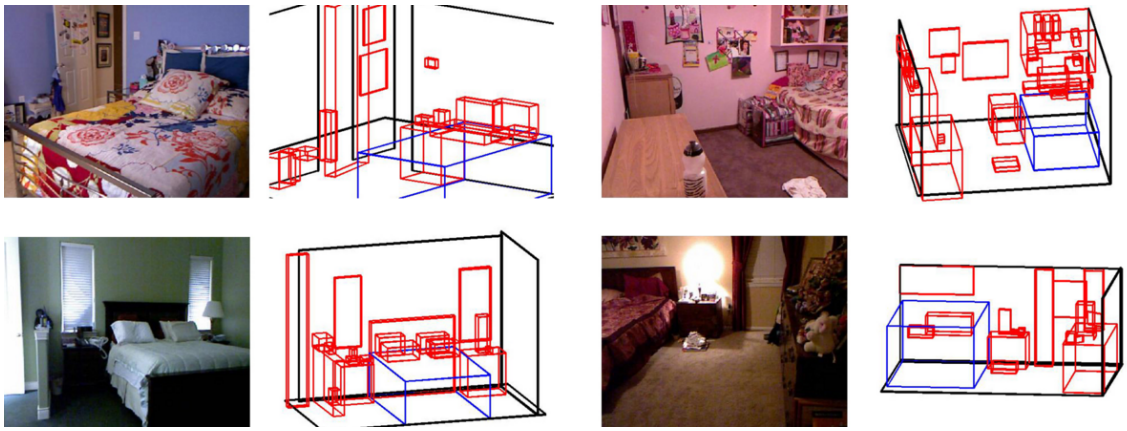

Figure 1.4: Representing objects in the scene using 3D bounding boxes [7].

1.3 Contributions and organization

In this dissertation, we are interested in instance-level representation of an indoor scene that reveals important and fundamental information of a scene. This dissertation is divided into the following chapters:

Chapter 2: The related work is discussed in terms of the existing research paradigms: the 3D representation of instances, the semantic labeling methods, the object recognition methods and the benefit of context information to boost the performance for each of these paradigms. Our research follows the flow of existing paradigms, dig out the relation between them and finally unite them as complete scene understanding.

Chapter 3: We introduce the object presentation in cuboids, a tight representation of volumetric occupancy with 3D rotations scene from a single-view depth map observation. Rather than the traditional voxel-based methods, we developed another low-level vision feature: the dominant planes. An algorithm is presented to show object local structures using tight cuboids by matching dominant planes. This research is expected to Cuboid detection was first studied in [8] where more useful information of an object (e.g., 3D orientation and dimensions) is provided compared with the traditional bounding box approaches. However, over-detection and miss-detection are often seen when there are many insignificant planar surfaces or the scene is too clut-

tered. In this work, we want to attack this problem in two ways. First, improve the quality of plane candidates for cuboid initialization by taking advantage of color and geometry features of each plane candidate, and then we propose a new local plane optimization algorithm to find the optimal parameters for each cuboid. This research is expected to reinforce the object presentation techniques such as point cloud or voxel method, and to support many object-level tasks for scene understanding.

Chapter 4: To acquire instance-level segmentation, we combine the cuboid generation method from Chapter 3 with data-driven learning-based methods to make 3D bounding box hypothesis. Specifically, we use the output (i.e., category-specific score-maps) from any deep network learned from semantic labeling, together with the object geometric information, to generate holistic object instance level segmentations in 3D. In addition to the conventional evaluation method that calculates the intersection over union (BB-IoU) between generated bounding box and the ground truth, we present visible-point IoU (VP-IoU) to accommodate indoor situations where heavy occlusion exists and object bounding boxes have estimated sizes beyond the visible range. Our method generates tight bounding boxes and has the potential to bring back the missing instances that were obligated by baseline detections. Our contribution has three-fold. First, it is the first step to fully combine the semantic labeling and instance segmentation to achieve complete scene understanding. Second, it develops the advantage of the intermediate layers of the fully-trained networks by applying the category-specific score-maps that are capable of handling a variety of indoor objects. Third, our method is compatible to boost the performance of any deep-learning network-based algorithms.

Chapter 5: The instance hypotheses are generated respectively per each category in Chapter 4. However, objects are placed in certain groups for similar functions. Thus, we introduce a new instance segmentation module together graphical model-based context information to directly find object boundaries. Our framework is sig-

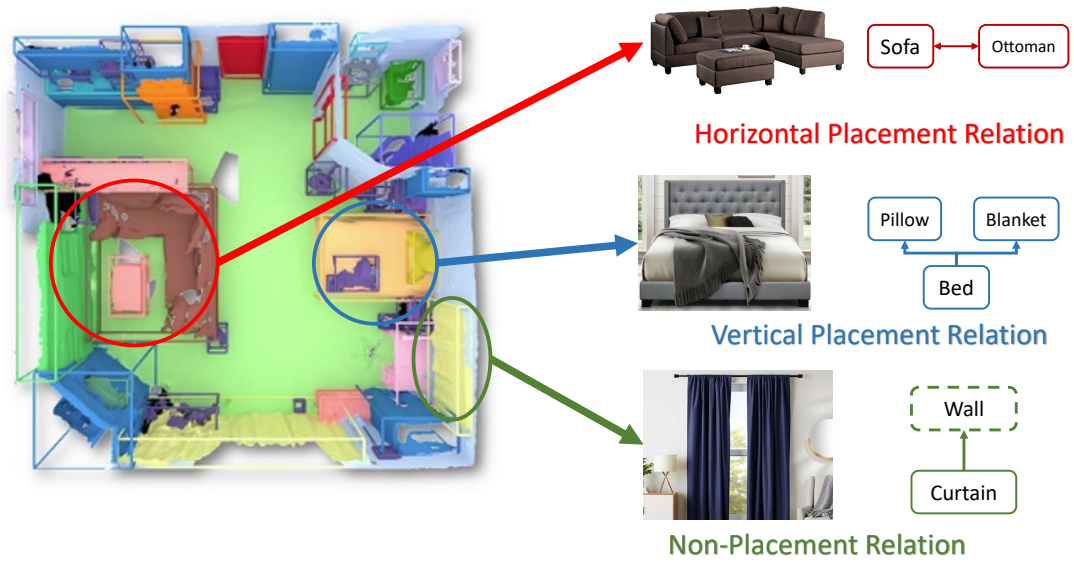

Figure 1.5: Indoor objects appear in different placement relations.

nificantly different from the existing deep learning-based approaches. Moreover, we are able to efficiently incorporate trained network outputs with non-network models (dual graphical models) to segment all instances with high objectness without relying on computational expensive instance-level network training.

Chapter 6: We try to push the boundary even further by combing the relation models we get in Chapter 5 with the instance bounding box generation method in Chapter 4 to build a complete flow to generate instance segmentations. We present a novel framework for holistic 3D instance segmentation using semantic labeling information together with graphical model-based context information to generate instance-level segmentations. We also expand the duo placement relations in Chapter 5 to get more complete trio-context models (vertical placement model, aka. VPM; Horizontal Placement Model, aka. HPM; Non-Placement Model, aka. NPM).

Chapter 7: Unlike the graphical model-based context information we use in the previous chapters, we aim to encode the context information using deep learning net-

works. We propose a re-inference framework to implement the high-level knowledge as supportive information for semantic segmentation. Given any semantic segmentation detectors, our method consider the objects' Local Collaborative Presence feature as guidance and re-train a segmentation module to acquire higher accuracy.

Chapter 8: We summarize all the findings and briefly talk about our next research: generate context models using deep learning methods to boost the collaboration between the semantic labeling module and the context information. In the preliminary work we present, the measure of "collaborative object presence (COP) is introduced and some simple evaluations are performed to show the effectiveness of it. Our future work is about applying the COP in a deep learning network frame to be trained for instance segmentation directly.

CHAPTER II

RELATED WORK

Core problems on scene understanding in the 3D space include semantic segmentation, object detection and instance segmentation. Nowadays, researchers are capable of achieving remarkable results in each of the problems. However, all these problems have been studied and achieved in separate algorithm flows rather than a united understanding scheme. Enabling machines to understand objects in 3D scenes as a whole is a fundamental necessity for many applications, such as autonomous driving, augmented reality and drone navigation.

2.1 3D object representations

With the popularity of various low-cost depth sensors, RGB-D images are often used to understand an indoor scene. Research on 3D understanding using RGB-D data has two main trends. Low-level processing usually focuses on the spatial capacity for object detection and representation [20–22]. High-level inference is to infer scene semantics by analysing of the geometry and structure of objects [23–26].

Bounding boxes (2D or 3D) can provide object-level understanding in the scene, which show not only the location and size of different objects, but also their orientation and 2D or 3D occupancy [**?**, 27, 28]. They can also provide a relatively holistic view of each object. For example, the visible parts of a bed include mainly a headboard, pillows, sheets and blankets. Instead of recognizing them one by one, we treat them as one bed object, collectively and holistically [27, 28]. However, when deep learning is applied for bounding box generation, the outputs are confident scores of

bounding boxes and box parameters [27, 28]. Bounding box representation cannot provide detailed label information where the number of distinct objects is usually much less than that in pixel-level labeling due to the occlusion and sparsity problem of depth data. Improving the quality of 3D point set is shown as helpful to improve bounding box generation, like [27], which requires the fully registered point cloud and identifies limited object categories.

Some researchers attempted to use some geometric primitives to provide an intermediate scene representation, including planes [24, 29, 30] or cuboid [8, 31], or other geometric constraints as prior shapes to represent indoor objects. However, the geometrical representation is often applied to the whole scene which makes it unable to show the object category labels or instance level object segmentation.

Traditionally, the bounding boxes are generated around potential objects for the latter object recognition task. In [32], the bounding boxes are optimized by aligning with gravity in an indoor scene. The bounding box-based algorithms aim at separating objects from background by displaying their spacial occupation and position [26]. Although the bounding box contains a possible object, it does not provide any geometrical or shape information about the object. To take advantage of object's geometry or shape as well as 3D orientation, a cuboid detection algorithm was proposed in [8] to detect objects from their visible planar surfaces. Different from bounding boxes, cuboids can rotate freely in 3D space or lie on the surface of an object, providing useful mid-level representation for semantic scene understanding.

The wide availability of consumer RGB-D sensors has boosted the research of object detection where color and depth are often used together due to their complementary nature. For example, the algorithm proposed in [33] detects and segments an object from color frames and then generates bounding boxes based on depth information. The depth information is not involved in the early state of object detection. It is our intention to use color and depth thorough the cuboid generation process.

The method proposed in [34] focuses on the segmentation of an 3D object from a RGB-D image by using strong shape priors learned from the mesh model of the given object type. The cuboid in this work is considered as a weak shape prior that can be general and flexible enough to handle most indoor objects with more or less planar surfaces.

2.2 Semantic labeling methods

Pixel-level labeling or semantic labeling illustrates semantically important details in a scene by providing the category label information for each pixel in an image [35,36]. With the development of GPU supported computation, deep learning networks has become the standard in computer vision tasks including semantic segmentation. The early approaches generate segmentation masks by classifying region proposals for saliency detection. Then it is developed to show the contours and areas of different objects in a scene. Driven by the powerful deep learning approaches, recent labeling algorithms can identify up to 40-50 object categories in an indoor scene [1,37]. Due to the fact that the computational cost for semantic labeling is very high, transfer learning is introduced to store knowledge while training for one problem, then apply it to a similar but different problem. By adapting a pre-trained network as initialization for several hidden layers, the training efficiency for semantic labeling using deep learning networks is boosted. However, pixel-level labeling cannot show instance level object segmentation. Cluttered objects that belong to one category would not be separated individually. For example, when a table is surrounded by chairs, semantic labeling is not designed to tell the range for each chair. The classification information for each pixel cannot directly support holistic object-level scene understanding. Thus, in addition to pixel level labeling, the widely used benchmark datasets [2,3] also provides instance level ground truth for segmentation as a high-level holistic task.

2.3 Object recognition methods

The object detection is to determine where and what objects are present in a scene. It is a challenging problem in image processing and computer vision because of large variation in object shapes and significant occlusion. With the development of deep learning [13, 14] and availability of RGB-D data, the performance of object detection has been improved significantly in recent years. However, challenges still remain to detect objects in both detailed and holistic ways.

Instance segmentation is more challenging than semantic segmentation as it requires the additional reasoning of objects. Instance segmentation methods can be categorized into two groups, proposal-based approaches and proposal-free approaches. Proposal-based approaches build systems upon object detection and append segmentation modules after bounding box proposals. Inspired by the recent success of Mask R-CNN [38] on 2D instance segmentation, 3D-SIS [39] develops a proposal-based system. GSPN [40] presents a generative model for generating proposals. Bounding boxes are effective and intuitive to show the 2D or 3D range for each object [27,35,41], but they also lack of details of the object boundaries. Researchers have been trying to use a cuboid-shaped boxes to represent objects and preserve the detailed object shape information at the same time [19,28]. A two-step approach was proposed in [28] that combines objectness estimation and object recognition for bounding box generation. Some approaches [19] generate 3D bounding boxes by removing out irrelevant 3D points according to re-projected 2D bounding boxes. The ground truth data of bounding boxes are provided independently with that of pixel level labeling, making them lack consistency and compatibility. Some studies tried to find an intermediate representation to present the scene both holistically and in a detailed way. Approaches include using planes [24, 29, 30], cuboid [8, 31] or other geometry primitives as prior shapes to represent indoor objects. On the other hand, proposal-free methods cluster points into instances based on the similarity metrics. Bounding boxes were created

from pixel level labeling for a fully registered 3D point cloud [27] with little occlusion. SGPN [42] trains a network to predict semantic labels, a similarity matrix between all pairs of points and point-wise confidence for being a seed point. Some researchers further develop the idea to investigate on the probability of adjacent 3D points belonging to the same object instance [43, 44]. However, the geometrical representation is applied to the point-level without indicating object categories or instance level segmentation. The process still uses limited holistic information.

2.4 Contextual modeling for scene understanding

There are two main paradigms for object detection: per-pixel semantic labeling and bounding box generation. The former one provides the spatial areas of different objects in an image, the latter one provides a holistic view with a set of cuboid-shaped boxes to represent the location, size and orientation of different objects in a 3D scene. Usually, more object categories are considered in pixel-level semantic labeling than those for 3D bounding boxes due to heavy occlusion and sparsity of depth data. Thus, the two tasks are often studied separately. There are some recent efforts to combine them in order to take advantage of their complementary nature for object detection [8, 19, 27].

Due to the limited information contained in each single pixel, efforts have been made to add holistic knowledge for semantic labeling [45]. For example, an image is segmented into equal-sized cells for labeling [46]. The normal distribution of depth data at the pixel level is used for object recognition [26]. The distribution along the gravity direction is considered during pixel level labeling in 3D space [1]. Other methods [47, 48] process 3D point cloud directly without voxelization. While showing promising results, these methods lack the ability of modelling geometrical structures of the input point cloud. A point cloud of an indoor space usually contains much more number of points, which means the network can only process a slice of the input point

cloud at each time, which disables global reasoning of the space. Recently, Graham et al. [12] propose an super-efficient volumetric CNN based on sparse convolution to process the entire point cloud of an indoor scene, which achieves promising results on the semantic segmentation task, it becomes practical to train networks with significantly more layers [49]. Due to the increased complexity of working directly in 3D, especially in large environments, many methods use some type of projection. In VoxelNet [50], the 3D data is first reduced to a bird eye view before proceeding to the rest of the pipeline. More recently, deep networks on point clouds are used to exploit sparsity of the data [12, 43, 44, 51].

Scene understanding usually involves holistic prior knowledge about the scene. For example, some methods focus on the perpendicular patterns of indoor rooms and furniture [23,30]. Some algorithms extract indoor structures to find the general room configuration [24,29,52]. To find and localize all objects in the whole scene, some predefined scene templates were used as prior knowledge for directed local search [53]. Using a graphical model, the spatial occurrence pattern among objects in 2D images is captured to improve the object detection rate collectively [17, 54].

CHAPTER III

OBJECT REPRESENTATION IN CUBOIDS

In this chapter, we will talk about the indoor object representation using cuboids. This is a basic approach of extracting low-level vision information from the raw RGB-D information. We find the traditional point cloud and voxels interpret the scene in a scatter way. Our cuboid approach is able to show the local structure in the view of 3D space. In this chapter, there're four sections: (1) the preliminary work; (2) local plane detection in RGB-D data; (3) Cuboid initialization and optimization; (4) experimental results.

3.1 Preliminary work

3.1.1 Point cloud representation

A lot of projects have been focused on point cloud to increase its accuracy, aiming to show more details [55, 56]; some other projects aim to regularize the point cloud from a big picture [23, 57, 58]. The mostly used assumption is the Manhattan assumption, which assumes that the big planes in an indoor scene probably follow one of the three major coordinates. Researchers used this assumption to segment the map of the indoor scene to get some understanding of it. However, the main problem for depth sensors, such as Kinect, is that the sensor can only show the appearances of scenes. People can understand scenes better not only because we know what is seen, but also because we can infer what is unseen, especially in the indoor scenes where many occlusions exist It is almost improbable for the point cloud to get every detail of the indoor scene. So our goal is to make the computer understand the scene by

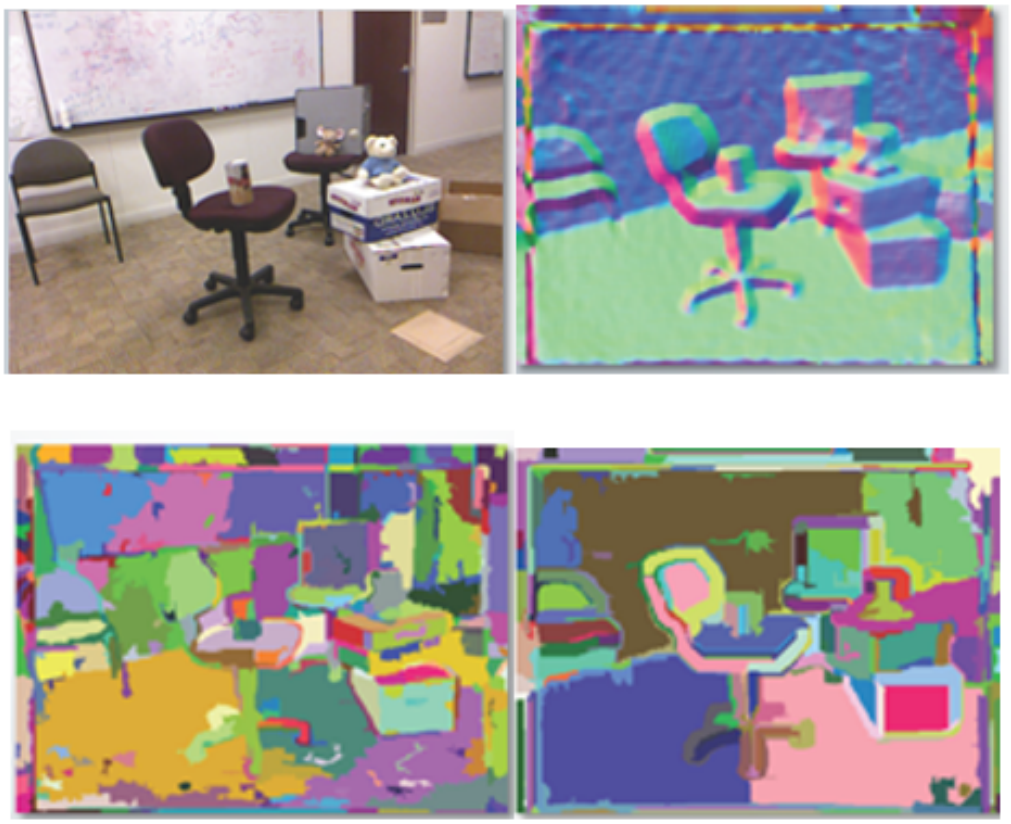

Figure 3.1: The preliminary work from [8]: Row 1 shows the color image, normal image with the three channels containing the x, y and z components. Row 2 shows the superpixels generated by using the normal image only and the superpixels generated by using both the color and normal images.

separating the objects from room structure and inferring the unseen behind the data points observed from the sensor based the Manhattan assumption.

Research on scene understanding using RGB-D data has two main trends. Low-level processing usually focuses on the spatial capacity for object detection and representation [20–22]. High-level inference is to infer scene semantics by analysing of the geometry and structure of objects [23–26]. In this work, we are interested in a mid-level representation of an indoor scene that reveals important and fundamental structures of a scene. Cuboid detection was first studied in [8] where more useful infor-

mation of an object (e.g., 3D orientation and dimensions) is provided compared with the traditional bounding box approaches. However, over-detection and miss-detection are often seen when there are many insignificant planar surfaces or the scene is too cluttered. In this work, we want to attack this problem in two ways. First, improve the quality of plane candidates for cuboid initialization by taking advantage of color and geometry features of each plane candidate, and then we propose a new local plane optimization algorithm to find the optimal parameters for each cuboid. This research is expected to support many object-level tasks for scene understanding.

3.1.2 Voxel representation

Traditionally, the bounding boxes are generated around potential objects for the latter object recognition task. In [32], the bounding boxes are optimized by aligning with gravity in an indoor scene. The bounding box-based algorithms aim at separating objects from background by displaying their spacial occupation and position [26]. Although the bounding box contains a possible object, it does not provide any geometrical or shape information about the object. To take advantage of object's geometry or shape as well as 3D orientation, a cuboid detection algorithm was proposed in [8] to detect objects from their visible planar surfaces. Different from bounding boxes, cuboids can rotate freely in 3D space or lie on the surface of an object, providing useful mid-level representation for semantic scene understanding.

The wide availability of consumer RGB-D sensors has boosted the research of object detection where color and depth are often used together due to their complementary nature. For example, the algorithm proposed in [33] detects and segments an object from color frames and then generates bounding boxes based on depth information. The depth information is not involved in the early state of object detection. It is our intention to use color and depth thorough the cuboid generation process. The method proposed in [34] focuses on the segmentation of an 3D object from a

RGB-D image by using strong shape priors learned from the mesh model of the given object type. The cuboid in this work is considered as a weak shape prior that can be general and flexible enough to handle most indoor objects with more or less planar surfaces.

3.2 Local plane detection in RGB-D data

Our approach has three stages to improve cuboid detection. First, plane candidate refinement is to improve the quality of plane patches by involving an additional split-and-merge operation. Second, dominant plane generation is to select major plane candidates according to their depth features. Third, cuboid candidates are initialized by dominant plane candidates and optimized by maximizing local fitness.

3.2.1 Plane candidate refinement

Initial plane candidates can be created by any segmentation (e.g., [9]) or clustering algorithm (e.g., K-means or Meanshift). The objective is to create a set of super-pixels as building blocks for cuboid generation. Similar to the implementation of [8], we use the efficient K-means algorithm followed by connected component analysis to create initial patches. The K-mean clustering is done in a 7D feature space where each pixel is represented by the concatenation 1D depth, 3D RGB, and 3D normal. In practice, it is possible a cluster may contain parts from multiple objects when they are adjacent, leading to mis-detected cuboids. We propose a plane candidate refinement technique to improve the quality of initial plane candidates by involving a split-and-merge operation. The idea is to split a cluster into multiple pieces if it is not flat enough (i.e., low consistence of normal) and then to merge them with adjacent clusters if they share similar normal. For each cluster representing a plane candidate,

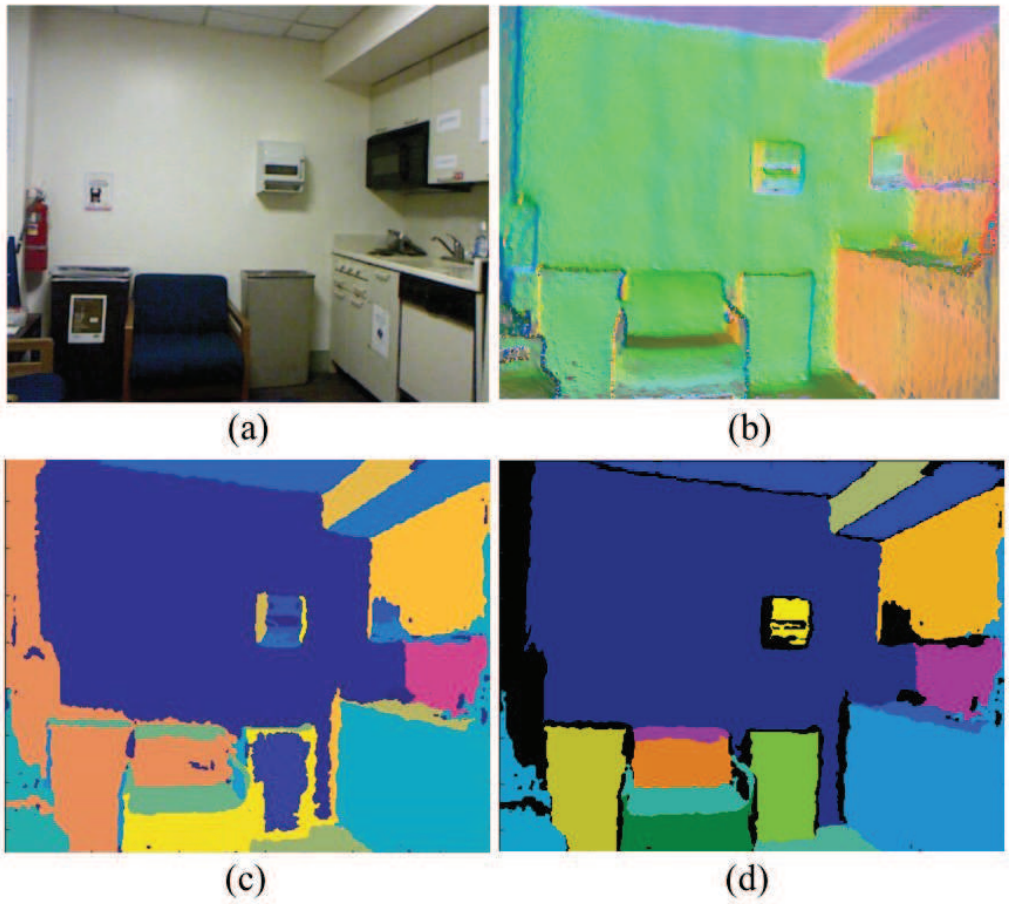

Figure 3.2: The dominant plane generation. (a) An input image; (b) The normal map; (c) Plane candidate generation [8, 9]; (d) Plane refinement + dominant plane generation (the pixels in black are ignored for cuboid initialization).

its flatness term is defined as:

$$F = \max_{d=1,2,3} \{\mathrm{V}ar(Pn^{(d)})\}, \tag{3.1}$$

where $Pn^{(d)}$ is the set of values of the dth dimension from the normal vectors in the cluster. We only perform the split and merge operation to those plane candidates with less flatness (e.g, below 20%). The split step is done by K-means clustering of 3D normal to reduce under-detection, and then a merge step is used to regroup newly generated clusters with adjacent ones if their mean normal are similar. This merge operation is necessary to avoid over-detection.

3.2.2 Dominant plane generation

We want to focus on dominant plane candidates for cuboid matching and optimization that are associated with major objects in the scene. Therefore, we develop a composite criterion to evaluate the significance of a plane candidate that jointly considers the flatness, spatial coverage and continuity. For each plane candidate P, its significance measure is evaluated as:

$$M = \alpha_1 * F + \beta_1 * G + \gamma_1 * H, \tag{3.2}$$

where F, G and H represents the flatness, spatial coverage and continuity, respectively, and α_1, β_1 and γ_1 are the weights to accommodate different scaling factors. G_i is the number of points in P_i and H_i is the continuity of P defined as:

$$H = \max_{p \in P}\{\vec{N_i} \cdot \vec{p}\} - \min_{p \in P}\{\vec{N_i} \cdot \vec{p}\}, \tag{3.3}$$

where $\vec{N}$ is the normal of cluster P and p is a point in P. H is used to penalize the case that a cluster contains multiple planes. The significance measure M is computed for all plane candidates and only those top ones for generating cuboid candidates. Fig. 3.3 shows an example of plane refinement and dominant plane generation where plane generation is improved in the areas of the arm chair and two trash cans.

3.3 Cuboid initialization and optimization

In our work, all cuboid candidates are initialized by dominant planes that have been verified to be associated with different major objects in the scene, such that we can lower the risks of over-detection and under-detection of cuboids. We follow the method in [8] to initialize the cuboid candidates by matching two nearby planes. Then we develop an optimization process to further improve the accuracy and local fitness of each generated cuboid. To do so, we define an objective function to reflect the local fitness of a cuboid candidate that can be optimized numerically.

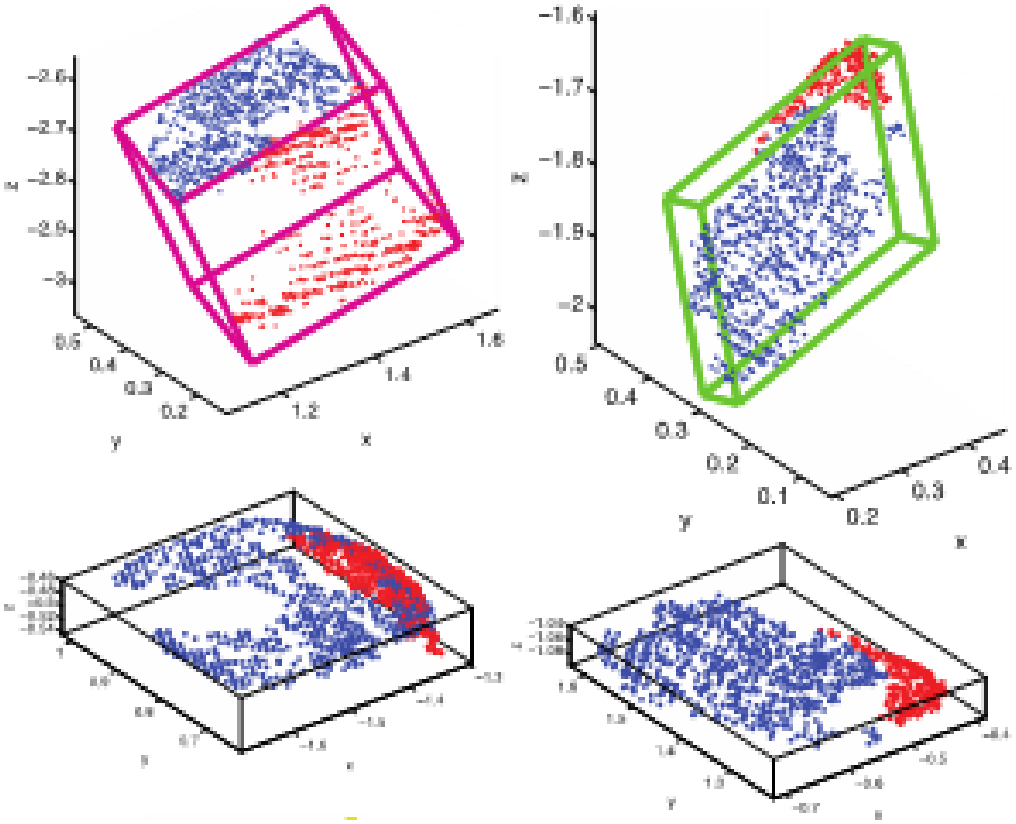

Figure 3.3: The cuboids are generated from two dominant planes.

Given a cuboid candidate, its local point set S is selected within the range of two scaled cuboid sizes (from $1-\sigma$ to $1+\sigma$). The cuboid parameter is denoted by $\mathbf{x} = \{x_c, x_r, x_d\}$, where x_c is the center position of the cuboid; x_r is the rotation matrix from the cuboid coordinate to the scene coordinate; x_d is the 3D dimension of the cuboid. The energy function $E(\mathbf{x})$ is defined to quantify the local fitness of the cuboid,

$$E(\mathbf{x}) = \alpha_2 * M(\mathbf{x}) + \beta_2 * V(\mathbf{x}) + \gamma_2 * O(\mathbf{x}), \tag{3.4}$$

where α_2, β_2 and γ_2 are weights to adjust the relative importance of three terms; $M(\mathbf{x})$ is the point fitness term that encourages all points in S to be close to the cuboid surface; $V(\mathbf{x})$ is the visibility term that penalizes the case if the cuboid lies in the space where is known empty; and $O(\mathbf{x})$ is the coverage terms which encourages data points to cover the most area of the cuboid surface. $M(\mathbf{x})$ can be computed

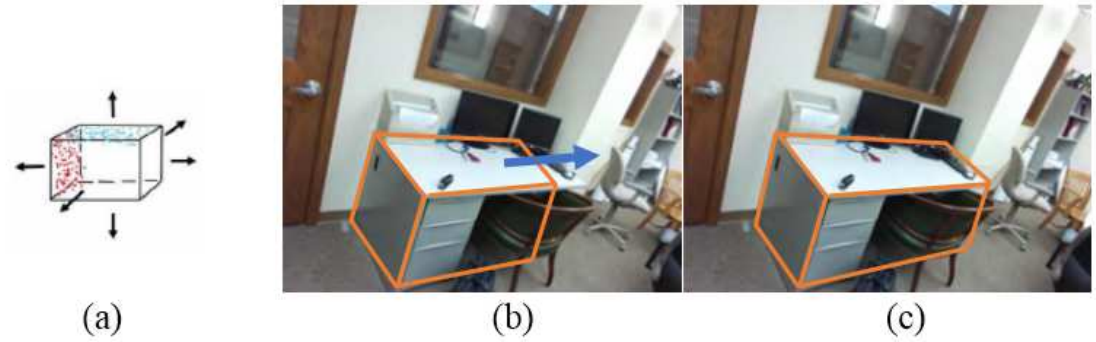

Figure 3.4: Cuboid optimization illustration. (a) All six directions are optimized. (b) and (c) show the results before and after optimization.

from S directly while the latter two occupancy-related terms can be computed via local voxel-based representation.

Specifically, the point fitness terms $M(\mathbf{x})$ is the mean distance from each point in S to the closest cuboid surface, defined as below,

$$M(\mathbf{x}) = \frac{1}{N_s} \sum_{p \in S} D(p, \mathbf{x}), \tag{3.5}$$

where N_s is the number of points in S and $D(p, \mathbf{x})$ is the distance of point p to the nearest surface on the cuboid parametrized by $\mathbf{x}$. To compute $V(\mathbf{x})$ and $O(\mathbf{x})$, we first create a local 3D volume along the cuboid orientation that encloses local point set S, and then we quantize that volume into voxels that are classified into three groups: surface voxels which have data points inside; unknown voxels which are block by surface voxels; and empty voxels which have to be transparent in order to not block the surface voxels. With the help of local voxel-based representation, the visibility term $V(\mathbf{x})$ is shown as,

$$V(\mathbf{x}) = \frac{W(\mathbf{x}, S)}{U(\mathbf{x})}, \tag{3.6}$$

where $U(\mathbf{x})$ is the total number of voxels in the cuboid and $W(\mathbf{x})$ is the sum of the numbers of surface and unknown voxels. The coverage term $O(\mathbf{x})$ is computed as,

$$O(\mathbf{x}) = \max_{c=1:6} \{R^c(\mathbf{x}, S)\}, \tag{3.7}$$

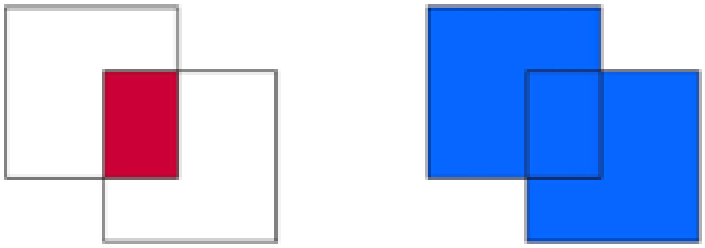

Figure 3.5: The Intersection (left) and the Union (right) of two boxes.

where c is the index of the six cuboid facets; $R^c(\mathbf{x}, S)$ is the percentage of the number of voxels on the cuboid's cth facet covered by the surface voxels in S.

Cuboid optimization is accomplished by maximizing the local fitness between the cuboid and local point set S. We invoke the heuristic direct search method [59] to optimize cuboid $\mathbf{x}$ by maximizing (3.4) numerically. Specifically, the cuboid orientation x_r is assumed to be fixed due to the fact that all cuboids are initialized by dominant planes with reasonable reliability and accuracy. Only the cuboid center x_c and cuboid dimension x_d are to be optimized. There are six facets of each cuboid each of which is associated with a scaling factor along the normal direction as shown in Fig. 2(a). For simplicity, the six scaling factors are treated independently, and each scaling factor is optimized sequentially and individually. After the optimization of six scaling factors, the cuboid center and dimension can be computed straightforwardly, as shown in Fig.2(b) and (c) where the size of the desk is more accurate after cuboid optimization.

3.4 Experimental results

The experiments were conducted on the NYU v1 Kinect dataset [7] that includes 1074 RGBD images with ground truth bounding boxes. In our experiments, we chose $\alpha_1 = 10$, $\beta_1 = -0.00001$, $\gamma_1 = 30$ for plane refinement, and $\alpha_2 = 100$, $\beta_2 = -1$

and $\gamma_2 = -1$ for cuboid optimization. Most existing object detection algorithms are based on training or involve additional knowledge, such as the spatial prior [60] or 3D shape priors [34]. We are interested the case where no training or high-level priors are involved, like [8]. In addition, we developed three implementations to show the usefulness of each step. The first one only involves partial local optimization (with only point fitness) without plane refinement. The second one has full local optimization without plane refinement. The third one is the complete algorithm. Specifically, we only focus six major indoor objects for performance evaluation.

We resort to a voxel-based scene representation to evaluate the performance of cuboid-based object detection. We first partition each scene into voxels. For each object, we find all visible voxels within the ground truth bounding boxes and those in the detected cuboid to calculate the IoU (intersection over union) ratio. A successful detection is declared if the IoU ratio of visible voxels is higher than a given threshold. The curves of detection rates under different IoU thresholds (from 0 to 1) of some major indoor objects are shown in Fig. 3.4. Our algorithms with cuboid optimization show significant improvement of over the baseline algorithm [8]. Specifically, plane refinement more helpful to those objects with major planes (desk and table), and the visibility/coverage terms are more useful for all objects due to the occlusion problem in the depth data. All algorithms drop quickly when the IoU threshold increases. This because the objects are usually cluttered in an indoor scene, and an object can be occluded by other ones or with some parts visible. To solve this problem, a training-based approach is necessary or high-level prior can be used to infer the occluded objects. Nevertheless, our cuboid-based approach provides an informative mid-level representation that is amenable to object-level vision tasks.

: the green dashed line: the results using [8]; the magenta dotted line: without plane refinement and partial optimization with only the point fitness term defined in (3.5); the blue dotted line: without plane refine and full optimization of local fitness

(3.4); Red: the proposed algorithm with plane refinement and full optimization of local fitness.

In addition to the six major objects, our approach is able to detect many other indoor objects that can be approximately by cuboids of different sizes. Some exemplar results of our algorithm and the one from [8] are shown in Fig. 3.7 for comparison. It is shown that the proposed algorithm is more reliable and accurate to detect those objects with planar surfaces. Our method also works in cluttered scenes, and the detected cuboids can generally tightly enclose the objects. However, not every major object (walls or floor) in the scene is detected. That is because that those objects only have one planar surface visible, while at least two visible planes are required to generate a cuboid.

3.5 Discussion

In [8], the computational load is mainly due to the massively generated cuboid candidates, most of which are duplications or very small. Plane refinement and dominant plane generation are helpful to initialize cuboids from significant planar surfaces representing meaningful objects in the scene. This process greatly reduces the number of cuboid candidates for local fitness optimization, and therefore the overall computational load of our algorithm is comparable with the baseline. The cuboid-based algorithms are effective for those objects with major planes but they may be limited to detect non-cuboid shaped objects. Also, occlusion may challenge the effectiveness of cuboid matching for object detection.

We have proposed three techniques to support robust cuboid matching for object detection in an indoor RGB-D image. The first to improve the quality of plane candidates according to a new flatness term. The second is to select dominant planes for cuboid initialization according to their significance measures. The third is to optimize each cuboid candidate with respect to three local terms. Significant improvements

are achieved compared with the baseline algorithm.

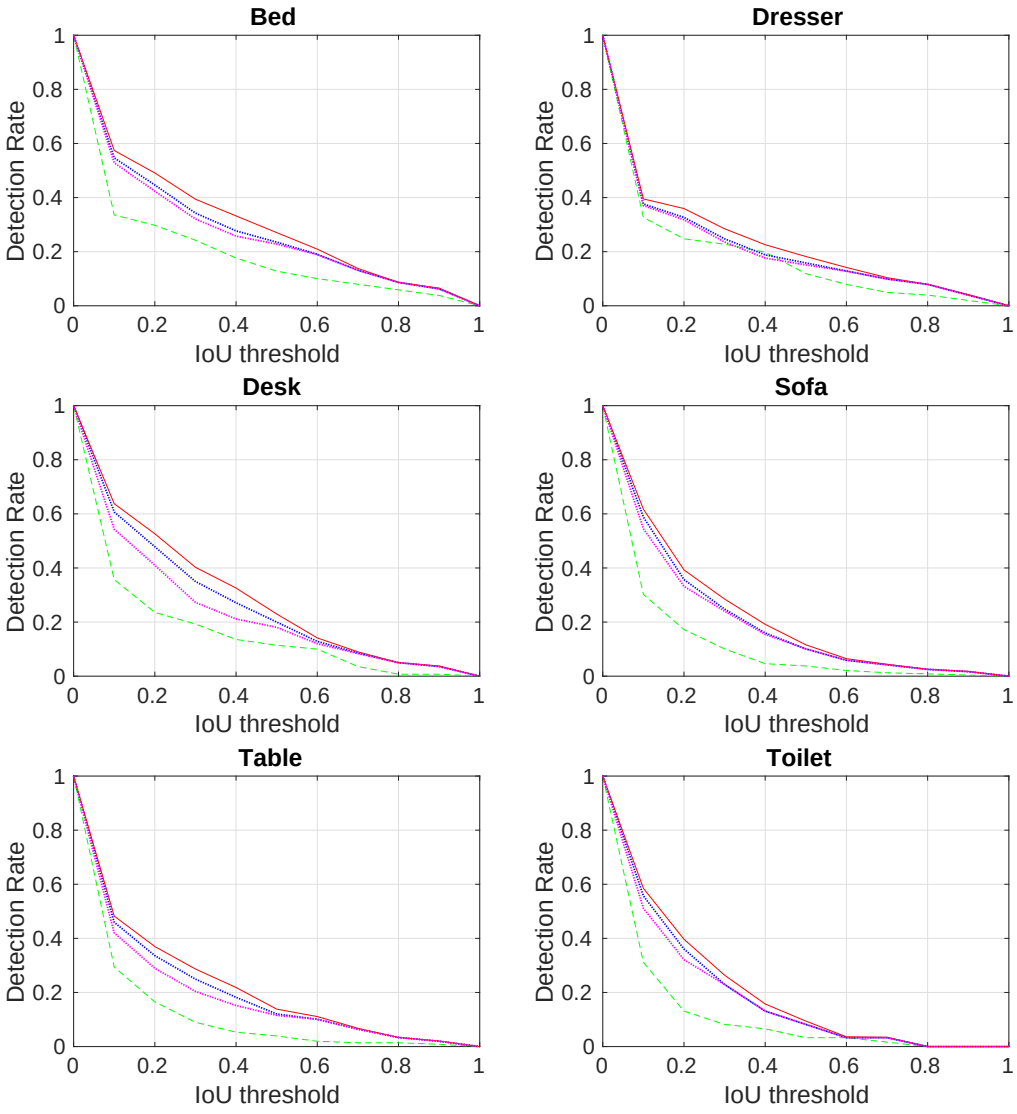

Figure 3.6: The detection rate of selected indoor objects under different IoU thresholds for different methods.

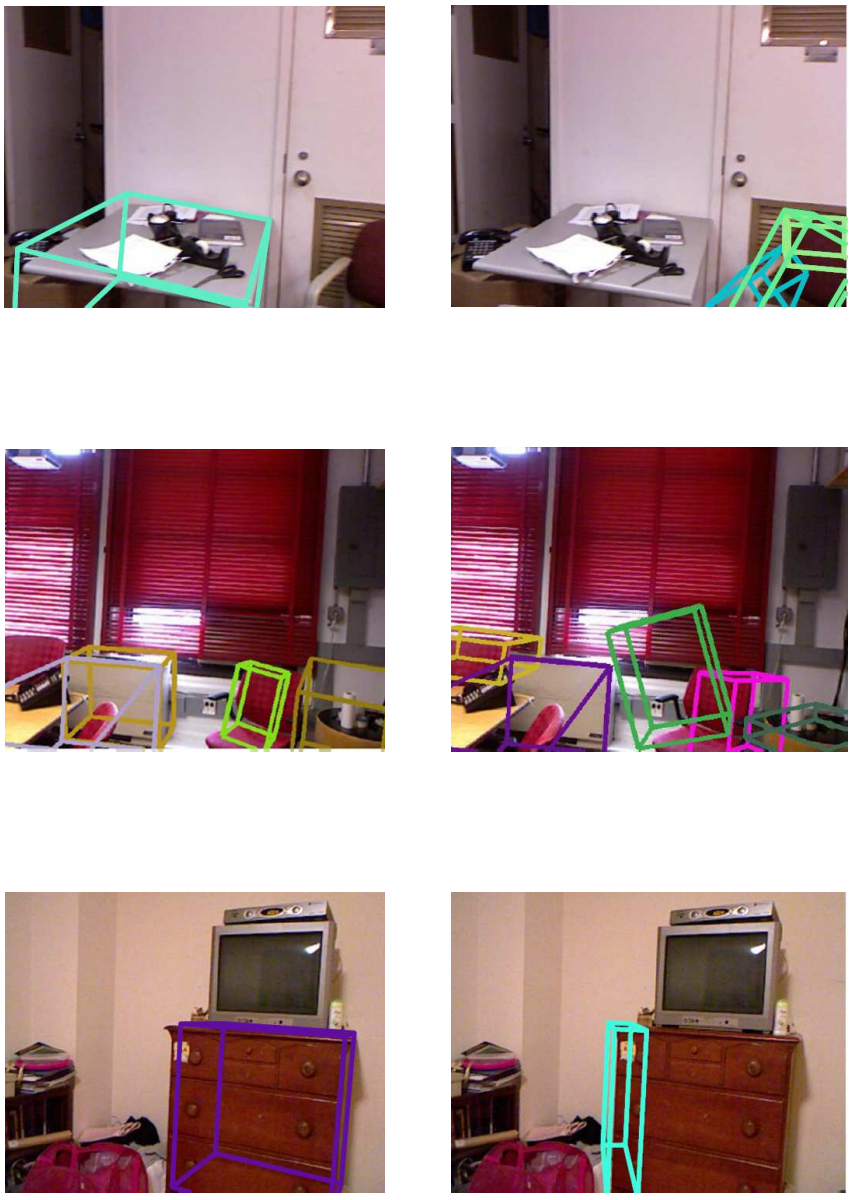

Figure 3.7: The comparisons of our method (column 1) with [8] (column 2).

CHAPTER IV

OBJECT REPRESENTATION IN BOUNDING BOXES

In this chapter, we discuss the indoor object representation using bounding boxes. This is the fundamental method of showing instances in the scenes. Our research is intended to take advantage of *semantics-rich* pixel-level labeling and *intuition-rich* bounding box representation for more complete scene understanding. Particularly, the recent rapid advancement in deep learning-based approaches provides a great opportunity to merge the two tasks in one flow with the goal of producing informative bounding box hypotheses for each category.

In this chapter, we have the following sections: (1) bounding box generation from pixel-level labels; (2) Objective function for our bounding box method; (3) optimization for bounding box generation and (4) experimental results and (5) Conclusion and discussion.

4.1 Preliminary work

The objective of our work is to generate the holistic bounding boxes from the detailed pixel-level labels. Thus, instead of using the results from the networks trained with the bounding box directly, we generate bounding boxes from the final classification output from the pixel-level label trained deep-learning network [1] as our baseline algorithm.

The baseline bounding boxes are generated by finding *compact* boxes with a minimum 3D volume that can contain all the classified points with respect to a target category. For category i, we define the corresponding A_i as the bounding box param-

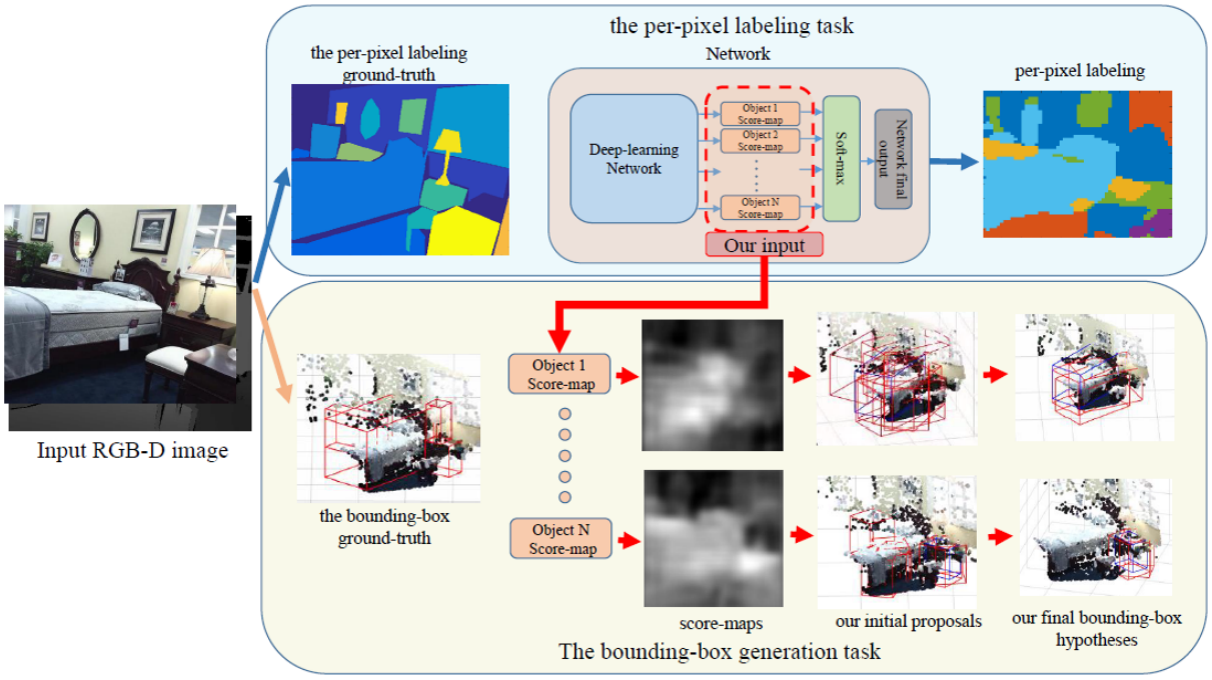

Figure 4.1: The flow of our bounding box hypotheses generation algorithm.

eter set from the baseline bounding box set $\boldsymbol{A} = \{A_i\}, i = 1, 2, ...n$, where n is the total number of categories. Same as the ground-truth, we assume the bounding boxes to be aligned with gravity. Thus, they have only one dimension rotation. Thus, we simply get the bounding box volume for each rotation angle r then get the coefficients from the range of the classified points C_i as shown in Fig. 4.2. The bounding box is generated using the equation below:

$$A_i = \underset{r \in \{0,\pi\}}{\arg\min} \{V(A_i^r)\}, \tag{4.1}$$

where $V(\cdot)$ returns the 3D volume of the input bounding box A_i with the rotation angle r with respect to the gravity direction. Since the score-maps from the deep learning network have relatively low resolution, we do not expect the rotation angle to be very accurate and r_i is optimized with a 5-degree increment in this work.

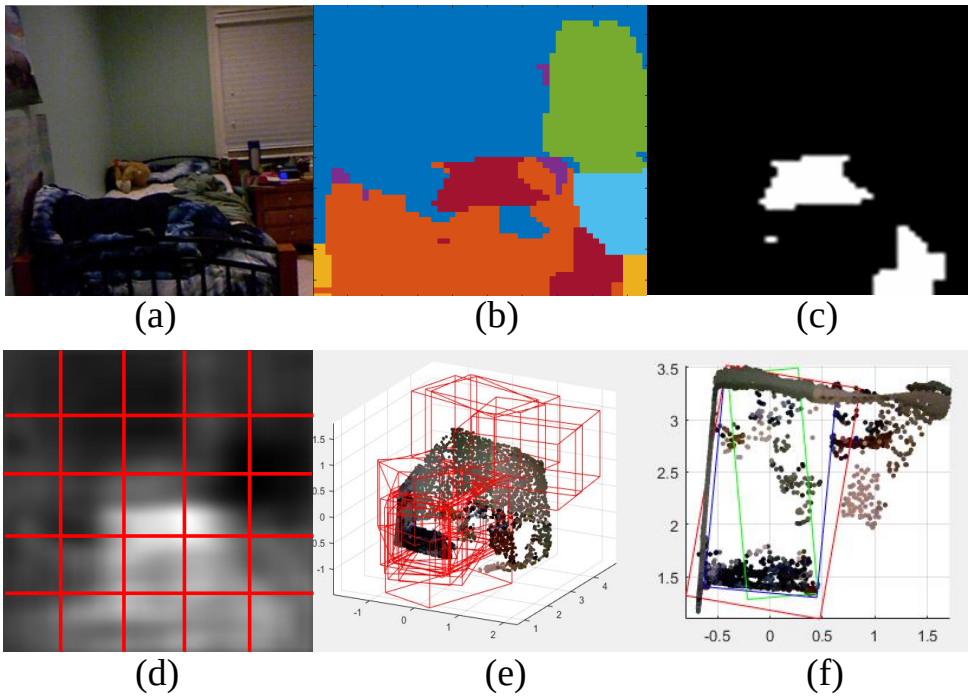

Figure 4.2: The comparison of the baseline and our approach. The baseline approach: (a) the testing image; (b) the network classification output from [1]; (c) the target object (bed) classified points C_i from (b); the final baseline bounding box A_i shown in green box in (f). Our approach: (d) the score-map ($Score(\cdot)$) shown together with the grid cells ($cell_j$); (e) all bounding box candidates generated from each $cell_j$; (f) our best bounding box hypothesis shown in red and the ground-truth bounding box shown in blue as a comparison.

4.2 Objective function

For each object category i, we define the corresponding B_i^k as the bounding box hypothesis k parameter set from the bounding box set $\boldsymbol{B} = \{B_i^k | i = 1, 2...n; k = 1, 2, 3...m\}$ in our results. m is the final number of hypotheses. The bounding box we generate should agree with both the deep-learning inference results and the visibility in the scene. To ensure the generated bounding box does not occlude the visible items in the scene, we quantize the 3D space into voxels and label all the empty ones as set T. Thus, for a bounding box candidate B_i^k, the evaluation function is defined as:

$$E_i^k = \alpha * F(B_i^k) - \beta * G(B_i^k, T), \tag{4.2}$$

where $F(\cdot)$ returns the average score of B_i^k from the score-map and $G(\cdot)$ returns the visibility penalty when B_i^k conflicts with the empty area T, as defined below:

$$G(B_i^k, T) = \frac{H(B_i^k, T)}{V(B_i)}, \tag{4.3}$$

where $H(\cdot)$ finds the amount of empty voxels T in bounding box hypothesis B_i^k; and $V(\cdot)$ returns the 3D volume of a bounding box. This indicates that for small objects, our method has relatively small tolerance to ensure the majority of the object volume to be solid. During the optimization process, E_i^k will be maximized by adjusting bounding box parameters in B_i^k. The benefit of using the visibility terms is shown in Fig. 4.3. $\alpha = 1$ and $\beta = 0.001$ are used in this work to balance the two terms in (6.12).

4.3 Bounding box optimization

Our optimization process has two steps: global rough search and the local fine-tuning. In the first step, to capture all possible bounding boxes in the scene, we initialize the bounding boxes in each of $S \times S$ (e.g., $S = 5$) grid cells defined on the score-map [46] using the mean coefficients of each object category in the training data, as shown in

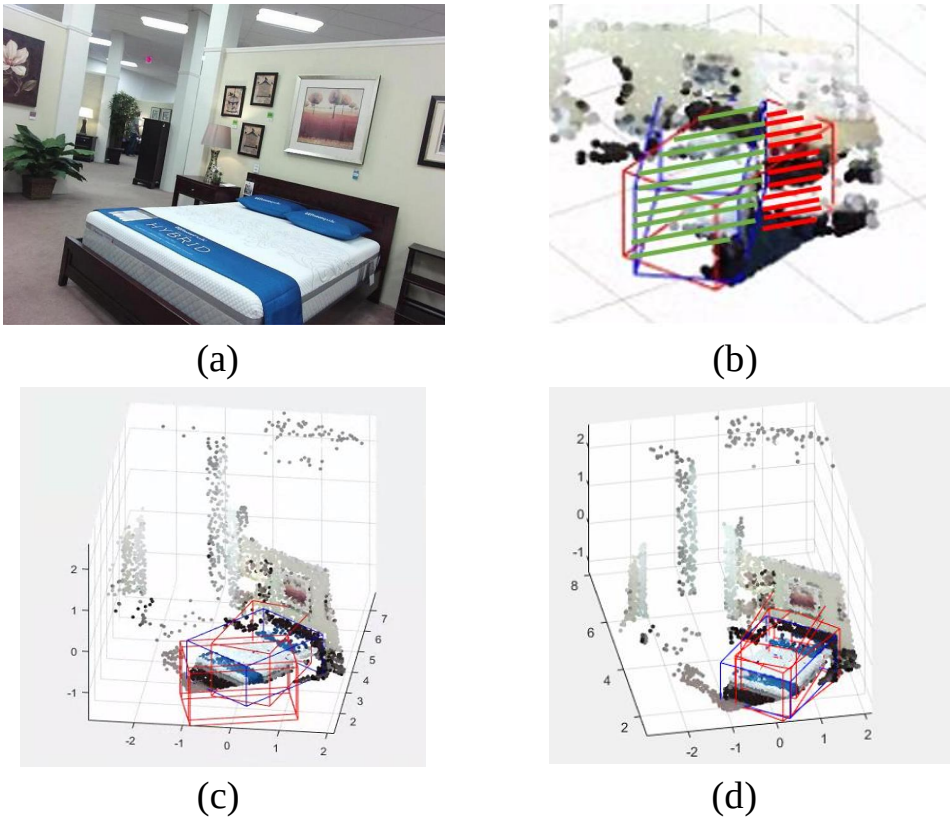

Figure 4.3: Illustration of the visibility penalty term. (a) A RGB-D image for a room. (b) The volume of a bounding box in red is divided into the visible (green lines) and invisible (red lines) parts by projecting each 3D point to the camera plane. (c) Top three bounding boxes generated without the visibility term. (d) Bounding boxes created with the visibility term.

Fig. 4.2. For each point z in a grid cell $cell_j$, the one with the highest network score ($Score$) is selected as the root point $Root_j$ as:

$$Root_j = \arg\max_z \{Score(z)\}, z \in cell_j. \tag{4.4}$$

A bounding box candidate is discarded if its score is too low. During optimization, the root point $Root_j$ must always stay in the bounding box to preserve the diversity in 3D space and reduce the computational load. In the second step, we optimize the 1D rotation along the gravity, center position and 3D dimension, a total of 7 parameters. The optimizer we use is a modified heuristic direct search method [59]. The direct

search optimizers are simpler and faster than gradient-based ones. We iteratively optimize the 7 parameters of each bounding box. Instead of using a fixed step size during optimization, the step size M is updated adaptively for each parameter as follows:

$$M_{new} = \frac{E_{new} - E_{old}}{|E_{new} - E_{old}|} * t_1 * M_{old} + t_2 * M_{old}, \tag{4.5}$$

where E_{new} and E_{old} are the evaluation score after and before a step move, and t_1 and t_2 where $t_1 + t_2 = 1$ are used to balance the intended step change and the original step size, respectively. In our experiment, we use $t_1 = 0.75$ and $t_2 = 0.25$. After optimization, the top three bounding box candidates are kept as our final results.

4.4 Experimental results

We conduct the experiments on the SUN-RGBD dataset [2] that includes 5285 training RGB-D images and 5050 test images with ground truth bounding boxes. Our algorithm is compared with the baseline [1] described in previous sections.

We evaluate our algorithm by two metrics. One is to use the traditional bounding box IoU (intersection over union) ratio (BB-IoU). The bounding boxes in our results are matched with the ground-truth ones. Our bounding boxes are generated from visible data points. However, the whole shape of the objects could hardly be seen in a single RGB-D image. The ground-truth bounding box provides the estimated full object size in 3D. Thus, part of the ground-truth bounding boxes cannot be obtained directly from the RGB-D image. To deal with this problem, we provide the visible point IoU evaluation (VP-IoU) which calculates the IoU only on the visible data points that are within the bounding boxes.

	BB-IoU			VP-IoU		
Category	*Baseline*	Ours-1	Ours-2	*Baseline*	Ours-1	Ours-2
Cabinet	7.43	11.68	**15.37**	21.53	32.31	**33.89**
Bed	24.81	**28.81**	28.08	51.42	54.14	**57.36**
Sofa	15.35	**21.99**	21.21	39.52	46.63	**47.4**
Table	13.97	**21.81**	21.61	28.1	37.35	**39.78**
Bookshelf	11.27	12.00	**12.3**	40.38	42.9	**45.65**
Counter	2.18	**17.91**	16.15	9.83	38.91	**41.28**
Desk	7.71	**17.24**	17.05	18.11	36.75	**39.91**
Dresser	6.6	12.62	**20.37**	20.25	31.97	**39.85**
Fridge	4.48	17.62	**19.82**	19.15	40.00	**43.97**
Sink	11.69	**22.42**	20.49	25.82	35.79	**37.39**
Board	5.15	**10.28**	10.06	32.78	33.61	**35.54**
Person	5.21	**8.71**	7.71	17.42	36.51	**39.04**
Toilet	27.35	**32.61**	31.36	44.96	47.97	**56.7**
Lamp	6.11	9.7	**10.59**	13.26	24.57	**26.96**
Nightstand	0	10.56	**11.64**	0.04	21.32	**23.44**
Bathtub	12.57	**23.87**	23.84	28.33	45.14	**50.79**
MEAN	10.12	17.49	**17.98**	25.68	37.87	**41.18**

Table 4.1: The evaluation results (%) in terms of both BB-IoU and VP-IoU for 16 major indoor objects, where *Base* is the baseline algorithm [1] presented in Section 3.1; $Ours-1$ is our method without the visibility penalty term (the second term in (6.12); $Ours-2$ is our final output both terms in (4.2).

4.5 Discussion

The quantitative results of BB-IoU and VP-IoU with respect to 16 major indoor objects are shown in Table 1. Ours-1 does not involve the visibility penalty term, whereas the Ours-2 method uses all terms in the objective function. We can find that both our methods outperform the baseline method in terms of both BB-IoU and VP-IoU. For BB-IoU, Ours-2 is comparable with Ours-1. It is understandable because BB-IoU is evaluated against with the ground-truth bounding boxes where the visibility term does not help much. For VP-IoU, Ours-2 is significantly better than Our-1 with a great margin. This indicates that Ours-2 can find much more meaningful visible points for the target category. It is worth mentioning that our method can even work well for those objects that cannot be handled by the baseline method (BB-IoU and VP-IoU are less 10%), such as *Nightstand* and *Counter*. Since all results are obtained by using the same score-maps as the baseline method, our experiment shows the deep-learning network actually extracts more meaningful information in the score-maps that is partially lost in the soft-max output.

Figure 4.4: Example images with annotation from SUN RGB-D dataset [2].

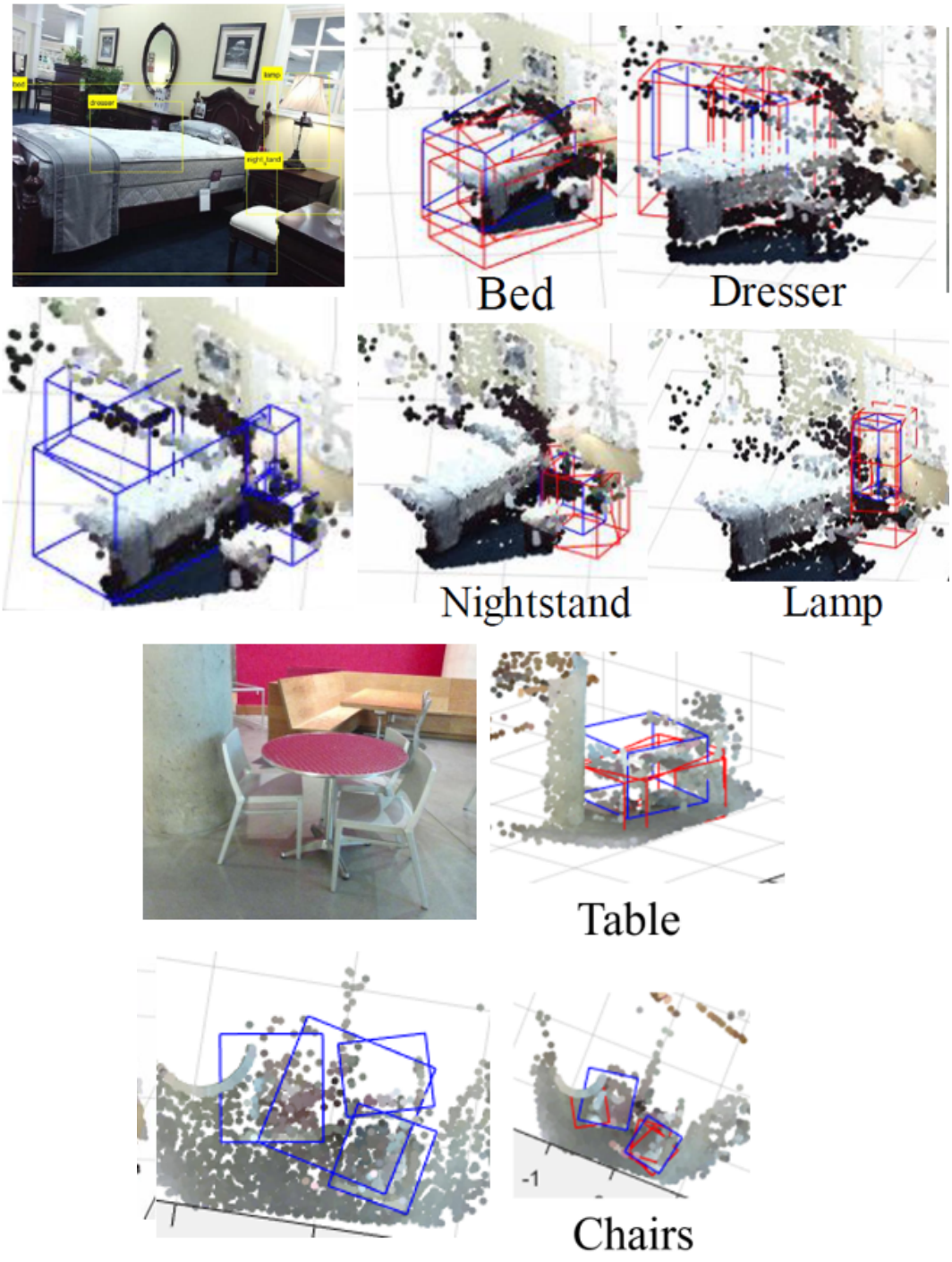

Figure 4.5: Some bouding box hypothesis generation examples.

CHAPTER V

RELATIONAL MODELLING OF INDOOR CONTEXT

In this chapter, we introduce a new instance segmentation module together graphical model-based context information to directly find object boundaries. Our framework is significantly different from the existing deep learning-based approaches. Moreover, we are able to efficiently incorporate trained network outputs with non-network models (dual graphical models) to segment all instances with high objectness without relying on computational expensive instance-level network training.

We organize this chapter as follows: (1) preliminary wok; (2) dual graphical models; (3) an application using the dual models: bounding box candidate generation; (4) experimental results and (5) Conclusion and discussion.

5.1 Preliminary work

Indoor scene understanding has been a challenging problem in computer vision because of large variation in object shapes and placement, and heavy occlusion and clutter. There exist three main schemes in scene understanding: per-pixel semantic labeling [35–37], bounding box generation [27, 28, 35], and scene level holistic understanding [**?**, 1, 8, 15–19, 30, 60]. The first scheme provides the contours and spatial areas of different objects and structures in an image. The second shows a set of cuboid-shaped boxes to represent different objects with certain size and orientation in . The third scheme includes various scene level tasks, including room type recognition [15, 16], scene structure classification [30] or other methods that incorporates high level knowledge from human understanding [1, 17, 18, 60]. There are two kinds

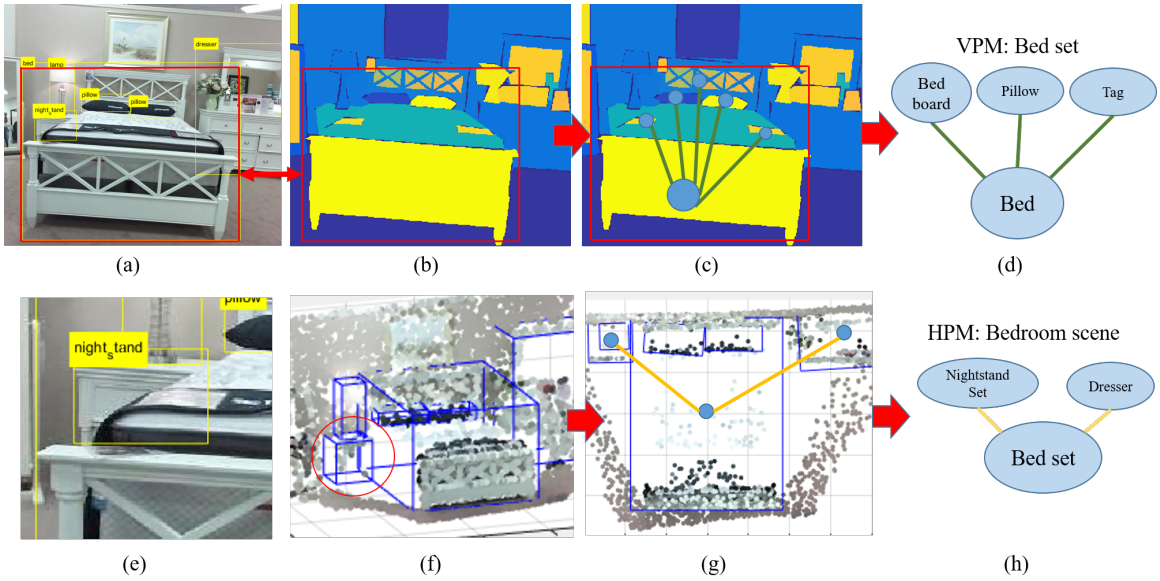

Figure 5.1: Illustration of VPM and HPM. (a) A bedroom image. (b) Ground-truth pixel level labeling of (a) where the bed is labeled as multiple items. (c) The vertical placement relationship of the bed set. (d) VPM for the bed set. (e) The cropped portion of the nightstand in (a). (f) Ground-truth bounding boxes in 3D space. (g) The horizontal object placement from the top-down view. (h) HPM for the bedroom.

of ground truth data used for training and validation, bounding boxes or pixel labeling. Usually, the former can be used to represent objects with relatively well-defined shapes, while the latter is more general and suitable for various objects or structures. Thus, more object categories are normally considered in pixel level labeling than those used for bounding box generation. On the other hand, there is a trend to combine both of them for scene understanding [8, 19]. However, there are some gaps between 2D pixel labels and 3D bounding boxes, both spatially and relationally.

5.2 Dual graphical models

Our objective is to integrate the three-level scene semantics in a bottom-up information flow where the two models, VPM and HPM, play complementary roles to bridge

the gap among three semantic levels. VPM serves as a *bridge* from pixel level labeling to bounding box initialization, and HPM plays as a *propagator* to use scene-level holistic configuration for collective bounding box generation .

There are two challenges in this research. The first one is about the placement between objects that often leads to occlusion and overlap problems, complicating bounding box generation. For example, a chair under a table is only partially visible and a pillow or sheet will cover part of the bed unlabelled. The second one is about the ambiguity and inconsistency of ground-truth data used for pixel labeling. There are two often-seen cases. The first is that different objects share the same label, for example, the nightstand and end-table were often considered to belong to the same category. The second is the same object was labeled differently. For example, some beds were labeled with a bed board, and some only include the mattress. Therefore, we involve two graphical models that capture the objective placement dependency to cope with those challenges with the aim to create reliable and accurate bounding boxes from inconsistent and ambiguous pixel-level labels. Traditionally a graphical model is generated considering the co-existence probability of objects as the edge weight S, as shown below.

$$p(n_1, ...n_k) = \prod_{i=1,2...k} S(n_i, M_i), \tag{5.1}$$

where

$$S(n_i, M_i) = p(n_i|M_i),$$

where n_i is the node for object i in the graph, M_i is the set of parents of node n_i, k is the total node number, and S stands for the edge weight calculated by the co-occurrence joint probability. In the following, we introduce the dual graphical models, VPM and HPM, that involve different weights as the closeness measure and similar training data.

5.2.1 Vertical placement model (VPM)

Given the ground-truth pixel labels and bounding boxes, we study vertical placement modelling by projecting all objects onto the ground plane from the top-down view. Then a 2D room layout is obtained by aligning all objects with gravity. Small ones are often placed on top of the bigger ones. Due to the fact that some small objects could be placed on different objects, we learn VPM with strong pair-wise connections by trimming off the weak ones [17,54]. In VPM, the nodes are object categories from pixel level labeling and the edges' weights are determined by the closeness measure that considers their co-occurrences and the overlap ratio in the layout view. Therefore, VPM is specified as,

$$g_{\mathcal{V}}(n_1, ...n_k) = \prod_{i=1,2...k} S_v(n_i, M_i), \tag{5.2}$$

where the edge weight is

$$S_v(n_i, m_i) = p(n_i|m_i) \times \frac{A(n_i)}{A(m_i)}, m_i \in M_i, \tag{5.3}$$

where $A(n_i)$ is the mean area of object n_i in the layout view. Thus, the edge weight S_v is calculated using the product of conditional probability and the 2D overlapping ratio between the two objects in layout view, which indicates the significance in the bounding box.

5.2.2 Horizontal placement model (HPM)

Similar to VPM, HPM is also learned from the top-down layout view of the projected 3D objects which embraces all object categories. The edge weight S_h is based on the co-occurrence probability of a pair of objects and the ratio of their center distance D with reference to their non-overlapping minimum distance B. For the objects without bounding boxes, the distance ratio is set to be one. HPM is defined as

$$g_{\mathcal{H}}(n_1, ...n_k) = \prod_{i=1,2...k} S_h(n_i, M_i), \tag{5.4}$$

where the edge weight

$$S_h(n_i, m_i) = p(n_i|m_i) \times \frac{D(n_i)}{B(n_i, m_i)}, m_i \in M_i, \tag{5.5}$$

which is the product of conditional probability and the 2D distance ratio in layout view between the center distance D and B. The non-overlapping minimum distance B is given by the summation of bounding box half size on the short side. Long distanced objects are encouraged due to the fact that distanced objects may exist in the scene outside of the image vision range, which lowers their presence probability.

5.2.3 Model learning

We learn the two models from the fully labeled dataset including two kinds of ground-truth data, i.e., pixel-level labeling and 3D bounding boxes. Specifically, the VPM learning involves both ground-truth data, whereas the HPM learning only uses bounding box ground-truth. We follow the framework in [54] and use the Chow-Liu algorithm [61] to maximize the likelihood of the training data (i.e., $g_{\mathcal{V}}(\cdot)$ for VPM and $g_{\mathcal{H}}(\cdot)$ for HPM). Firstly, the algorithm computes edge weights (S_v and S_h) as the mutual information for each object pair. Then, it finds the maximum weight spanning tree with the calculated edge weights. This algorithm only keeps strong pairwise information to generate an undirected graphical model.

5.2.4 Inference and implementation

Figures 5.2 and 5.3 illustrate VPM and HPM, respectively, which are learned from the ground-truth data (pixel labeling and bounding boxes) in the SUN-RGBD dataset [2]. The two models are versatile for different scene analysis tasks. (1) According to VPM, indoor objects can be classified into three groups, ground-level *base objects*, *accessory objects* placed on a base object, and stand-alone individual objects. (2) We can find the co-existence and exclusiveness between every object pair in both VPM and HPM.

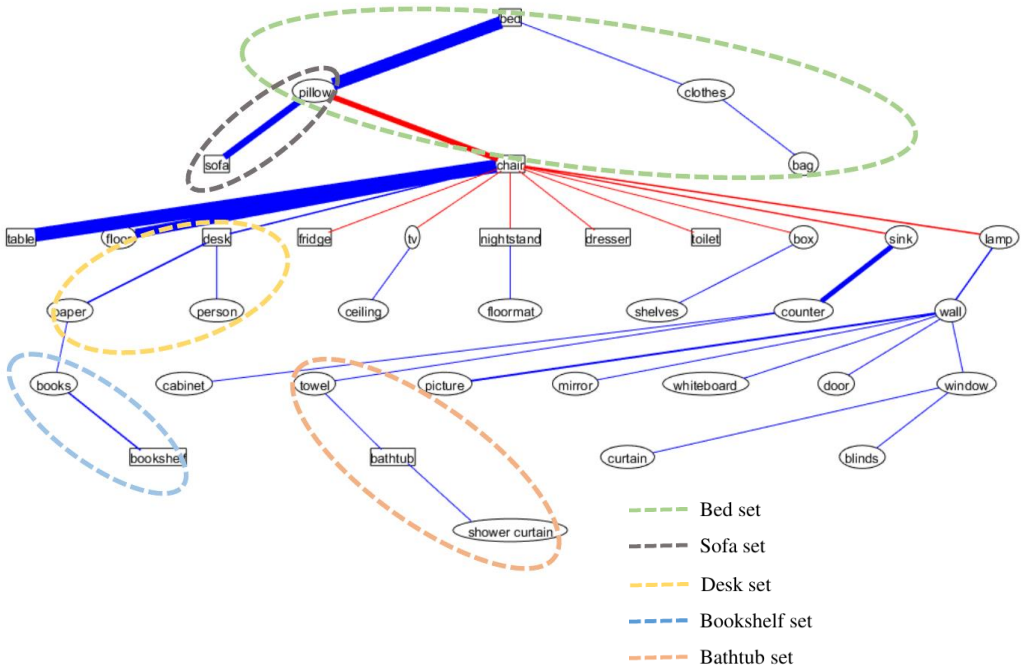

Figure 5.2: The vertical placement model of dataset SUN-RGBD [2]: on the top is the full model, the bottom shows the object sets in the model. The blue links shows the closeness while the red ones refer to exclusiveness. The thickness of links indicates the relation strength. The on-ground objects with bounding box tags are shown in rectangles while other objects are shown in ovals. The object set are rectangle-oval connections which stands for the base-accessory object relation. Note that the object relations are learned from training data. The dashed ovals are added only for illustration.

(3) The grouping effect in two models indicate different object sets and room types (denoted by dash ovals). (4) VPM and HPM can be used to refine and rectify ground truth data where the inconsistency and ambiguity may impede training and testing.

5.3 GM for bounding box generation

As a case study in this work, we will apply VPM and HPM to create 3D bounding boxes for all objects from pixel-level labeling results obtained from any deep learning algorithm. The VPM and HPM work together to bridge the gap between pixel-level

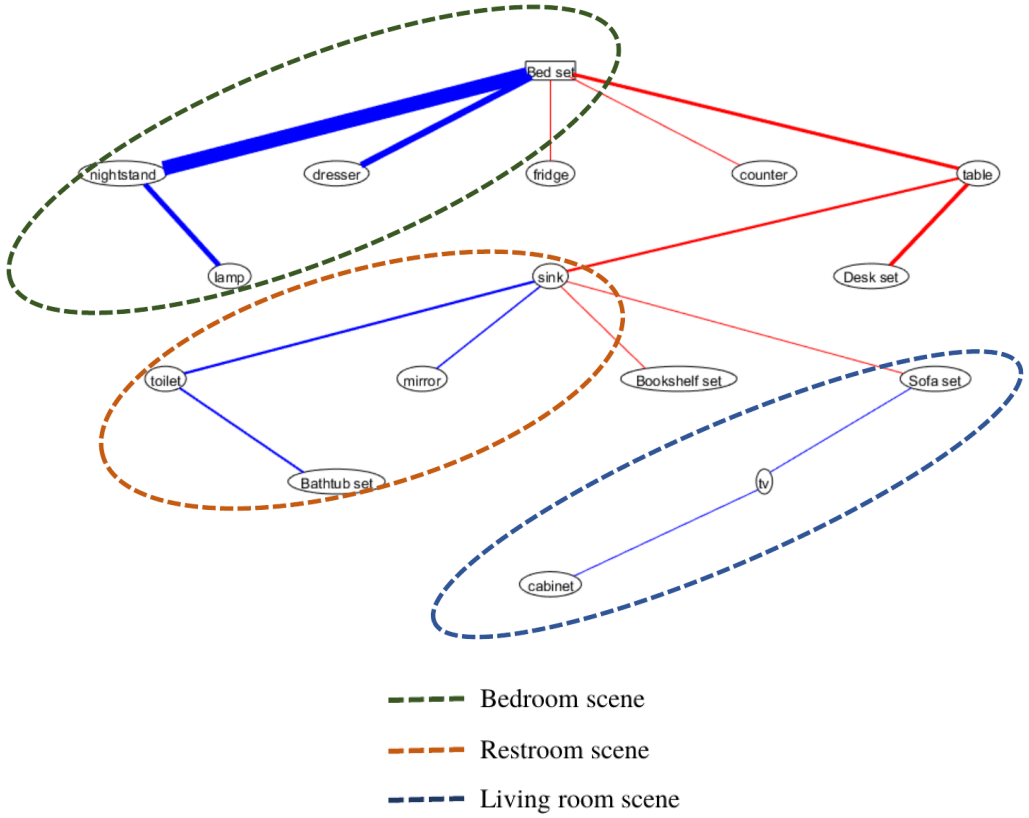

Figure 5.3: The horizontal placement model of dataset SUN-RGBD [2]: the full model is shown on the top. The bottom defines the scene groups in the model. The blue links shows the closeness while the red ones refer to exclusiveness. The thickness of links indicates the relation strength. The accessory objects are contained in the sets found using VPM as shown in Figure.5.2. The model automatically generates three object groups for bedroom, restroom and living room scenes.

labels and object-level bounding boxes in a sequential manner.

5.3.1 Bounding box initialization

Given a pixel-level segmentation map, we transfer the class label to the RGB-depth data points and then the labeled 3D points are projected from a top-down view to form a layout map, represented by L, similar to the way we created training data for VPM and HPM. For a specific given object, the size of the bounding box is obtained from the range of data points labeled as that object. Thus, we need only to find the bounding box orientation. We simply generate the bounding boxes for all directions with a step size of 5 degrees. Then, the bounding box with least points from other categories is considered to be the tightest and is selected as our initial bounding box. Given the layout view points set L, the 2D bounding box $\hat{X}_i$ is generated as:

$$\hat{X}_i = \underset{r \in \{0,\pi\}}{\arg\min} \{E(X_i(r) \mid L)\}, \tag{5.6}$$

where $X_i(r)$ is the bounding box parameter set for object i with orientation r in the layout view L. The bounding box evaluation function E gets the total point number that falls within $X_i(r)$ but not classified as object category i in the layout view L. To get the 3D bounding box, we add the height to X_i using the highest (along the gravity) labeled data point in X_i. Here we consider three kinds of objects, base objects, accessory objects and individual objects as defined before. Although pixel level labeling provides more object categories, we only consider the objects with ground truth bounding boxes.

5.3.2 VPM for base objects

VPM is used to re-label each given base object by finding its accessory objects to get a re-labeled layout view map. Following the baseline method in Section 4.1, the

bounding box $\hat{X_i^v}$ is generated as:

$$\hat{X_i^v} = \arg\min_{r \in \{0,\pi\}} \{E(X_i^v(r) \mid L_i^v)\}, \tag{5.7}$$

where L_i^v is generated from L after relabelling it with respect to the object i. L_i^v is obtained from VPM represented by $g_{\mathcal{V}}(...)$ defined in (2) by relabelling all accessory objects to be the base object underneath. The final bounding box heights are determined from initial labeled 3D points.

5.3.3 HPM for individual objects

In order to create bounding box generation for individual objects, we need to minimize the penalty from two kinds of uncertain 3D points during optimization. The first includes those with exclusive object labels as specified by HPM. The second corresponds to those with a low confidence sore as indicated by the segmentation map from the deep learning network. In other words, these two kinds of 3D points could be in a bounding box for any category. Hence, individual objects could have more flexibility in rotation r and dimension d during optimization, resulting in more accurate bounding box generation as:

$$\hat{X_i^h} = \arg\min_{r,d} \{E(X_i^h(r,d) \mid L^h)\}, \tag{5.8}$$

where L^h is created from L by suppressing the two types of uncertain points. Note that HPM is used at the scene level, which means L^h is generated for each image while L_i^v is generated for each base object.

5.4 Experimental results

Although VPM and HPM could be applied to various scene analysis tasks, we tested them for bounding box generation from pixel level labeling (Section 4). We used the SUN-RGBD [2] dataset that provides 5285 training images and 5050 testing images. The ground truth labeling provides a 37 classes set for pixel level labeling.

To generate the baseline bounding boxes, we use the pixel level labeling map from a recent deep learning method [1] as the input for the algorithm described in Section 4.1. The widely used evaluation measures mean average precision (mAP) under certain IoU threshold. We expect this evaluation could show more details about the real object area detection. We use two metrics to evaluate our method: the bounding box intersection over union (BB-IoU) and the visible point intersection over union (VP-IoU). The BB-IoU measures if the bounding box could be correctly found. In indoor scenes with heavy occlusion and sparse 3D data points, some of the bounding box ground-truth are inferred from the visible area. Thus, we use the VP-IoU measure to show if the visible point of the target object could be found. Note that the in VP-IoU, all the points in the target object bounding box are regarded as the same label as the bounding box, regardless their ground-truth pixel level labeling.

In Table 1, we quantitatively show that our method can improve the bounding box accuracy in both BB-IoU and VP-IoU compared with the baseline algorithm denoted as *Base* (Section 4.1). In these experiments, not all base objects have related accessory objects. Thus, some results in VPM are about the same as in the *Base* column. Significant improvements in VPM can be found in object categories "bed" and "sofa" because of the prevalent co-existence of bed-pillow and sofa-pillow, as shown in Figure 6.3. The scores for bookshelf are also improved thanks to the help of the inclusion of the book category. The bounding scores increase in general after we apply dual models because HPM provides more rotation flexibility for all objects. It is worth mentioning that the baseline method cannot correctly detect the nightstand class. After applying the dual models, part of the nightstand is recovered with the help of its related objects (mostly the bed).

Some qualitative results are shown in Figure 5.4 where we show the effect from VPM and the dual models (VPM+HPM). It is shown that VPM is able to help the inclusion of accessory objects to the base object, leading to improved bounding

boxes generation of base objects. Also, the dual models can assist bounding boxes for individual objects by including more uncertain points according to co-existence and exclusiveness encoded in VPM and HPM.

	BB-IoU			VP-IoU		
Category	*Base*	*VPM*	*Dual*	*Base*	*VPM*	*Dual*
Cabinet	7.43	7.43	7.66	21.53	21.53	21.76
Bed	24.81	27.67	28.81	52.13	52.79	54.29
Sofa	15.35	16.07	16.67	40.12	38.86	40.2
Table	13.7	13.7	14.7	28.79	28.79	30.43
Desk	7.78	9.74	10.14	18.09	19.02	19.68
Nightstand	0	0	4.8	0.04	0.04	10.4
Bathtub	12.57	13.14	13.46	28.33	28.63	29.22
Bookshelf	11.27	11.68	12.55	40.34	37.52	40.95
Toilet	27.35	27.35	27.85	44.96	44.96	44.78
Fridge	4.48	4.48	6.53	19.15	19.15	22
Dresser	6.6	6.6	7.86	20.12	20.12	20.1
Mean	11.94	12.53	13.73	28.5	28.31	30.35

Table 5.1: The quantitative results (%) in terms of both BB-IoU and VP-IoU for 11 indoor objects, where the baseline (*Base*) is compared against VPM and the dual models.

5.5 Discussion

It is worth noting that the ground-truth data of pixel-level labeling and bounding boxes still have some inconsistency and ambiguity which may complicate quantitative analysis. Thus the major bottleneck is the pixel-level deep learning algorithm that provides the input for our bottom-up flow. There are three possible directions that would enhance the strengths of VPM and HPM to improve the quality of bounding box generation. First, in stead of using the classification map, confidence maps for

each object offer more potential to improve the quality of bounding boxes. Second, we could enhance two models by incorporating more prior regarding the size and shape to improve the inference and optimization of (5.7) and (5.8). Third, VPM and HPM can be jointly used to improve the quality of ground-truth data in both training and testing data which consequentially manifest the contribution from two graphical models.

We have presented dual graphical models for relational modelling of indoor object categories, i.e., the vertical placement model (VPM) and horizontal placement model (HPM). Specifically, the former captures the co-existence of major and accessory objects, while the latter encodes ground level spatial configuration of different individual objects. The two models allow us to bridge the gap among the three levels of semantic scene understanding. As a case study, we apply the two models in a bottom-up flow to create object-specific bounding boxes in 3D space that are more informative and intuitive where the input is the pixel-level label result from any deep neural network. Experimental results show the promise of dual graphical models to improve the quality of bounding box generation. It is foreseeable the two graphical models can be used in other holistic object-level and scene-level analysis tasks.

Figure 5.4: From the results of VPM (top), from left to right, the figures are: (1) Two RGB images, (2) classified *bed* points from the deep network [1], (3) the *bed set* points after using VPM, (4) the generated bounding boxes (blue: ground truth, green: baseline, red: ours). From the results of dual models (VPM+HPM, bottom), from left to right, the figures are: (1) two RGB images, (2) white pixels classified *dresser* (the third row) and *nightstand* (the fourth row) from [1], (3) the uncertain points added to *dresser* (the third row) and the uncertain points that are exclusive with *bed* and added to *nightstand* (the fourth row), (4) the generated bounding boxes (blue: ground truth, green: baseline, red: ours).

CHAPTER VI

CROSS DOMAIN INSTANCE SEGMENTATION

In this chapter, we further expand the idea of co-existence of indoor instances, enrich the dual model we described in chapter 5, together with the bounding box optimization method we proposed in chapter 3 and 4, propose an end-to-end instance segmentation algorithm that works for both RGB-D images and 3D maps.

We organize this chapter as follows: (1) an overview of our approach which crosses the semantic labeling domain to instance segmentation domain; (2) relational modelling for indoor objects; (3) Instance segmentation; (4) Experimental results and (5) conclusion and discussion.

6.1 Preliminary work

Inspired by previous work on semantic understanding, we propose the relational graphical models to support deep neural networks for holistic instance segmentation. Compared with other approaches that are either for semantic labeling or for instance segmentation, our method investigates object-object relationships in terms of co-existence and exclusiveness among different objects, which indicate specific spatial indoor configuration. Therefore, we prevent the over-fitting problem in dealing with the 7-D input data (RGB-D+3D location). Our approach has the potential to improve the performance and functionality of many deep learning networks that are trained for pixel-level scene understanding.

Using RGB-D images as input information, the objective of instance segmentation is to classify each object and generate a mask in the form of a bounding box to localize

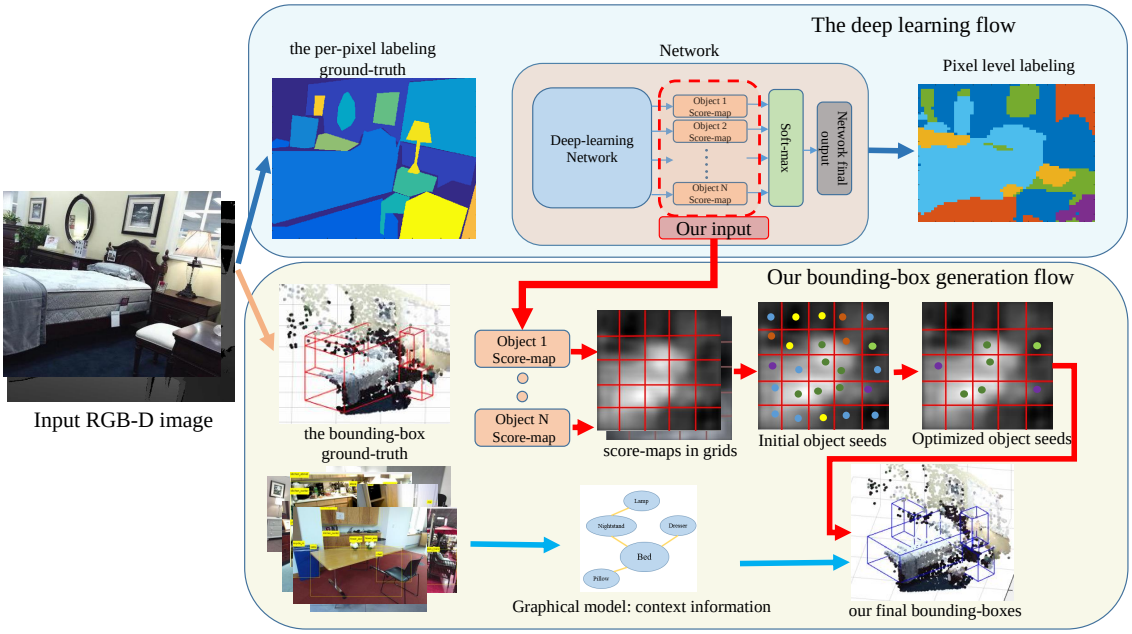

Figure 6.1: The proposed algorithm takes the score-maps from the deep network learned from pixel-level labeling as inputs to generate instance segmentation (red flow) together with the help of context information from tree-based models, which contain the VPM, HPM and NPM.

it. The depth data are represented as the point-to-plane distance in the camera coordinates. The camera parameters are known so that we are able to map each pixel in the RGB image to the 3D space. With the semantic labeling methods provided pixel-level per-class score maps, we can get the prediction distribution in the 3D space. Our method takes advantage of those score maps from semantic labeling, together with the object co-existence context, to gain holistic understanding and generate masks using 3D information. Fig. 6.1 shows the flow of our proposed algorithm. The process modules in the figure are introduced as follows.

Score Map Extraction We get the score maps for each object category from any well-trained neural network for semantic labeling. As the dashed red square shows in Fig. 6.1, the score maps are extracted before the soft-max layer to preserve the

original evaluation results.

Relational Model learning Using the location information given by the instance segmentation ground truth, we train our tree-based relational model which has three components: Vertical Placement Model (VPM), Horizontal Placement Model (HPM) and Non-Placement Model (NPM). Each of them describes a specific pattern of how objects are related in the scenes. The three relational models are jointly trained, then separately trimmed and applied. Similar to the traditional graphical model, our model is defined as the set of object mutual co-existence probability $R = \{r_{ij}|i, j = 1, ..., C\}$ as edge weights where C is the number of object categories. The weight R_{ij} for an object pair (n_i, n_j) is defined as follows:

$$R(n_i, n_j) = p(n_i, n_j)\Gamma(n_i, n_j), \tag{6.1}$$

where the $p(n_i, n_j)$ is the co-existence probability of two objects and the weight function $\Gamma(n_i, n_j)$ differs for the three components based on the object labels of n_i and n_j: the VPM R_V, the HPM R_H and the NPM R_N. We will introduce each model specifically in the next section.

Seed Point Calculation For the instance segmentation process, we find the object seeds first, then generate a full mask from it. The grid method is used following the idea in [46] to regulate the object density. Based on the fact that indoor objects are normally vertically aligned, the top-down view is used in our method to simplify the task. Specifically, we transfer the class label to the RGB-depth data points and then the labeled 3D points are projected from a top-down view to form a layout map, represented by L. We initialize the segmentation mask (cuboid-shaped boxes) in each of the $f \times f$ grid cells. Then, for each grid cell, the one with the highest network score is selected to represent that cell. Thus, we get the initial object seed set Q which contains the category information and the score value for each cell. Note that Q changes when L changes during our re-inference process.

Relational Model Re-Inference We apply the model by maximizing the scene-

level co-existence probability. Given a relational model R (HPM or NPM), and the initial object seed set Q, we infer the selected seeds S by optimizing the scene-level compatibility. It filters out unlikely seeds from Q via the relational constraint R. The updated S can be further applied to enhance the scope map L under the contextual influence of R to bring up new cell seeds for object categories not found yet. After the iterative optimization process, the selected seed set S is converged to describe the scene holistically.

Segmentation Local Optimization We optimize an object mask B_i for each final seed s_i found in S according to its local fitness while ensuring the seed point s_i is inside of the mask. The optimization is done in the layout map L and the mask is in the form of rectangles. At last, heavily overlapped bounding box candidates are eliminated to get the final results.

6.2 Upgraded relational modelling

In this section, we introduce the three relational models to adjust the co-existence probability of object pairs, i.e., the Vertical Placement Model (VPM), the Horizontal Placement Model (HPM) and , the Non-Placement Model (NPM) and how they are learned to represent 3D context information considering all objects from pixel-level labeling results obtained from any deep learning algorithm. Then, we introduce how the context model is trained and how the re-inference is done. In this work, the context-based graphical model works as prior knowledge to boost the performance of instance mask generation in the presence level. Therefore, we only consider the object existence in this section without considering objects' dimensions. Specifically, the VPM helps to merge the minor object classes to major objects for a holistic view to boost the layout score-map L. The HPM considers the object-object coexistence in the top-down layout view. The NPM (R_N) works to improve the identification of objects that are not placed with gravity, i.e. the hangable objects, by analyzing

their coexistence with the closest wall. The three models are initialized with edge weights for each object pair. Then the weak edges are trimmed off during the training process, rendering the models showing major co-existence information for our next re-inference process.

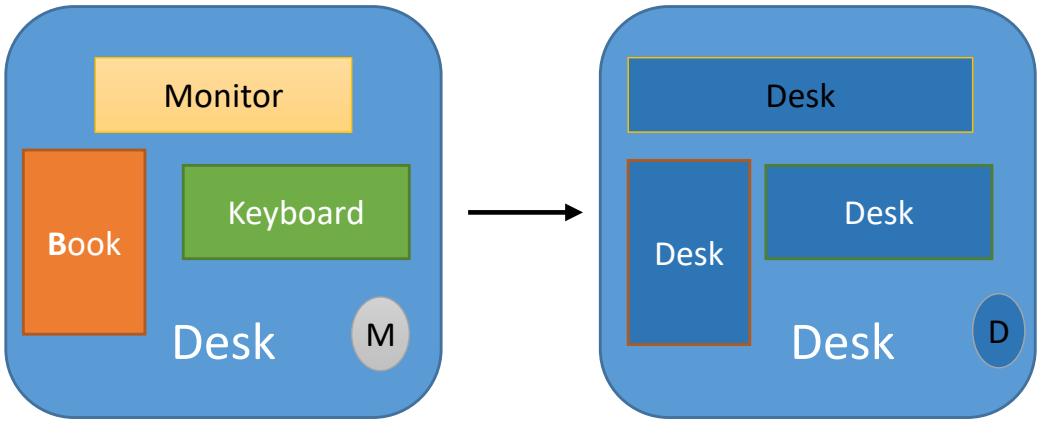

Figure 6.2: In the VPM, minor (small) objects are used to boost the detection of the occluded major object that appears under them. As shown in the figure, the monitor, the book, the keyboard and the mouse(M) are used to find the desk.

6.2.1 VPM for base objects

Given the ground-truth pixel labels and bounding boxes, we study vertical placement modeling by projecting all objects onto the ground plane from the top-down bird's eye view L, resulting a 2D room layout. Small objects are often placed on top of the base ones. As the example shown in Fig. 6.2, we could use the detection of small objects (monitor, book, keyboard and mouse) to enhance the presence of the desk. Due to the fact that some small objects could be placed on different objects, we learn the VPM with strong pair-wise connectivity by trimming off weak edges [54]. In the

VPM, the nodes are object categories from pixel level labeling and the edge weights are determined by the co-occurrences and the overlap ratio between two objects in the layout view. Therefore, for each object pair (n_i, n_j) that n_i is placed on top of n_j, the VPM-aware pairwise weight is specified as,

$$\Gamma_V(n_i, n_j) = \frac{A(n_i \cap n_j)}{A(n_j)}, \tag{6.2}$$

where $A(\cdot)$ is the area of an object in the layout view and the edge weight of VPM is represented as the product of joint probability and the 2D overlapping ratio between the two objects in layout view:

$$R_V(n_i, n_j) = p(n_i, n_j) \cdot \Gamma_V(n_i, n_j). \tag{6.3}$$

A learned VPM example is shown in Fig. 6.3 trained from the SUN RGB-D dataset [2]. The blue edges show the co-existence between objects, while the red edges mean exclusiveness. The thickness of both blue and red edges represent the relation strength. Our model extracts the vertical relations from the dataset without any artificial prior knowledge, and it shows consistency to people's common sense, for example: a pillow on top of a bed, a towel on top of a bathtub. Note that the strong link of chair-table and chair-desk is because the chairs are often placed partially under the table.

6.2.2 HPM for individual objects

Similar to VPM, HPM is also learned from the top-down layout view of the projected 3D objects which embraces all object categories. The edge weight R_H is based on the co-occurrence probability of a object pair and their adjacency. The pair-wise adjacency between two objects is measured by the ration between their actual center distance $D(\cdot)$ and the shortest center distance when they are placed next to each other $D(\cdot) - G(\cdot)$ (where $G(\cdot)$ is the gap distance along the center-to-center line), as

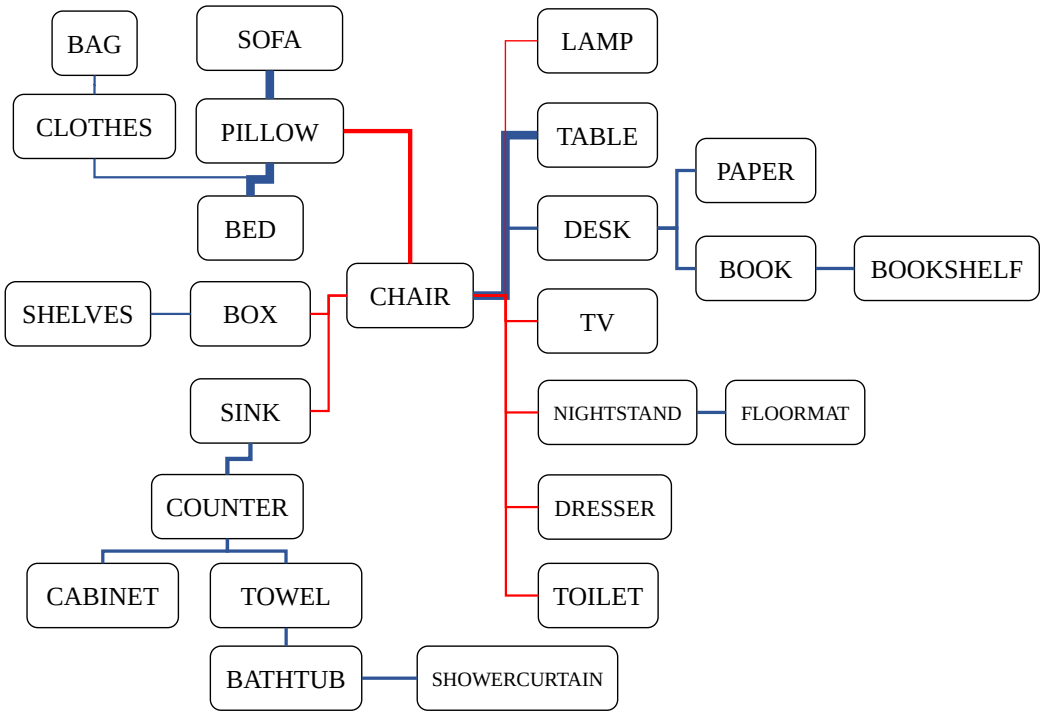

Figure 6.3: The vertical placement model of dataset SUN RGB-D [2]: The blue edges show the closeness while the red ones refer to exclusiveness. The thickness of edges indicate the relation strength. We use all the objects to train the model, then the weak edges are trimmed off. Isolated categories are not shown in the figure. This figure is graphically reproduced for better visualization.

shown in Fig. 6.4. Then the HPM-aware weight of a object pair (n_i, n_j) is defined as:

$$\Gamma_H(n_i, n_j) = \frac{D(n_i, n_j)}{D(n_i, n_j) - G(n_i, n_j)}, \tag{6.4}$$

which is intended to enhance the presence of a distant object due to the fact that its major portion is likely to be outside of the viewing range. For the objects without bounding boxes, the distance ratio is calculated using the pixel-level classification ground-truth. Then the edge weight of HPM is:

$$R_H(n_i, n_j) = p(n_i, n_j) \cdot \Gamma_H(n_i, n_j). \tag{6.5}$$

As shown in Fig. 6.4, we can see that due to the limited camera field of view, only a corner of the dresser is seen in the image which lowers its presence score, but the proposed HPM can boost its presence by using other frequent co-existent objects, such as the bed or the night-stand. A trained HPM example is shown in Fig. 6.5 using SUN-RGBD dataset [2]. Same as Fig. 6.3, the blue and red edges are for co-existence and exclusiveness and the thickness represents the relation strength. Our model extracts the horizontal relations among the objects. It automatically groups objects together based on their co-existence, which actually reveals the scene type. As shown in Fig. 6.5, the upper-left is the bedroom scene (bed, nightstand, lamp, dresser); the bottom is the restroom scene (toilet, mirror, bathtub, sink); the bottom-right is the living room scene (sofa, TV, cabinet). Thus, the scene understanding from our relational model is consistent with human understanding.

6.2.3 NPM for hangable objects

Unlike VPM or HPM, which are showing the object-object relationships, NPM considers the hangable objects on the wall. These kinds of objects normally are ignored in either VPM or HPM. Moreover, it is not easy to segment them out since they have very thin thickness. Thus, we introduce the help of the wall to enhance the detection of these kinds of objects. Given all the wall W in the layout map, a NPM is introduced

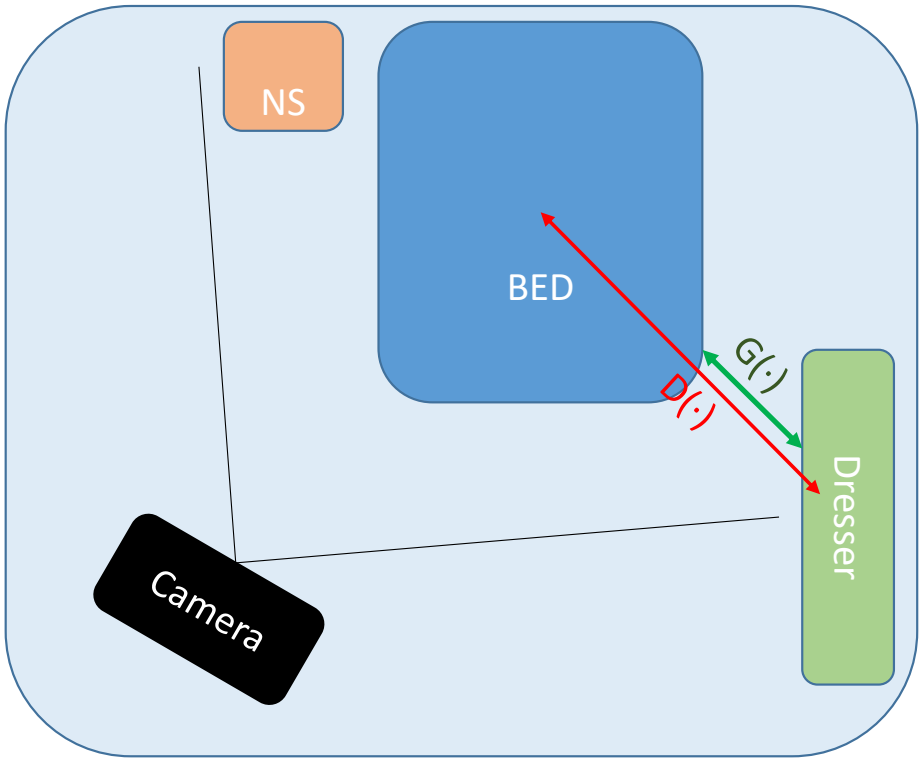

Figure 6.4: The HPM can boost the co-existence of two objects in the scene by the ratio between their actual center distance ($D(\cdot)$)) and the shortest possible center distance ($D(\cdot)$-$G(\cdot)$, where $G(\cdot)$ is the gap along the center-to-center line between two objects, i.e., the bed and the dresser.

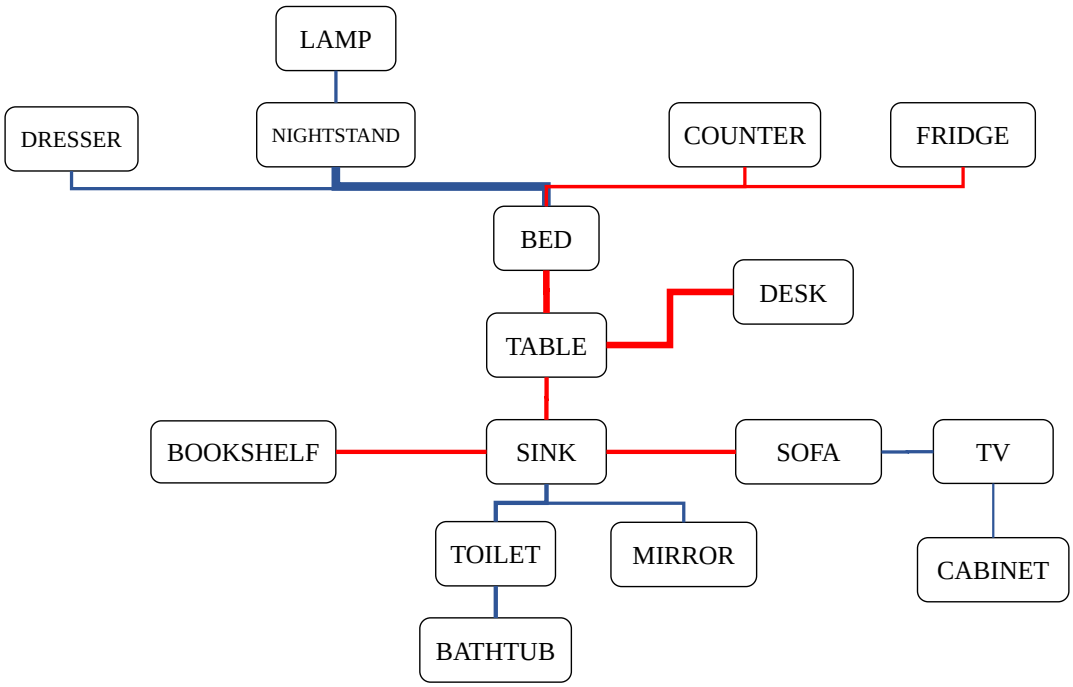

Figure 6.5: The HPM learned from the SUN RGB-D [2]: The blue edges show the closeness while the red ones refer to exclusiveness. The thickness of edges indicate the relation strength. Based on the VPM, we remove the minor objects (those on top of other objects) and keep only the major objects (those placed on the floor) for training. Then the weak edges are trimmed off. Isolated categories are not shown in the figure. This figure is graphically reproduced for better visualization.

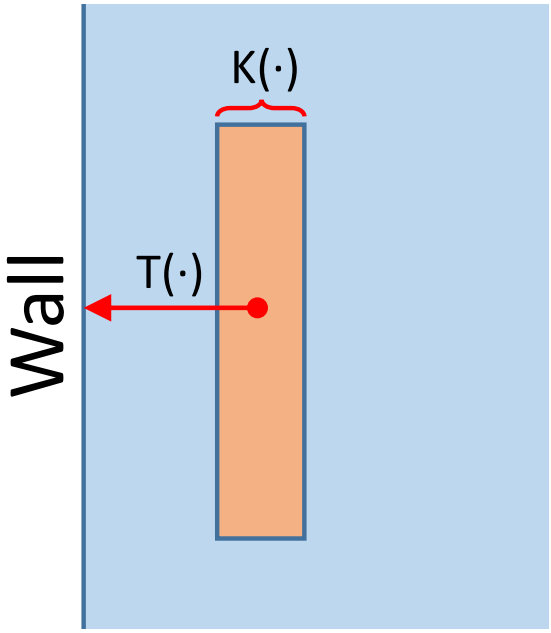

Figure 6.6: The NPM is focused on the thin hangable objects each of which is characterized by $T(\cdot)$ (the distance to the wall) and $K(\cdot)$ (the object's thickness along the wall-normal direction).

for any object category that appears next to the wall. NPM is also learned from the top-down layout view and checks for the wall in four major directions in the scene. The edge weight R_N is based on the co-occurrence probability of the object-wall pair and related to the object-to-wall-distance $T(\cdot)$ and object thickness $K(\cdot)$ from the top-down view. The NPM-aware object-to-wall weight is defined as:

$$\Gamma_N(n_i, W) = \frac{1}{T(n_i, W) \cdot K(n_i)}, \tag{6.6}$$

where $T(\cdot)$ returns the point-to-plane distance from the object center to the wall and $K(\cdot)$ is the thickness of the object along the $T(\cdot)$ direction as shown in Fig. 6.2.2. Therefore, the weight of each object-to-wall edge in the NPM is obtained as follows:

$$R_N(n_i, W) = p(n_i, W) \cdot \Gamma_N(n_i, W), \tag{6.7}$$

which is intended to boost the presence of the hangable objects according to their thickness and distance to the wall. In Fig. 6.7, we show a trained NPM model using SUN-RGBD dataset [2] that shows the relationship between each hangable object and the wall. During re-inference, the recognition of hangable objects could benefit from the wall that is relatively easy to detect.

6.2.4 Model learning and re-inference

We follow the Chow-Liu algorithm [61] to learn three relational models in an efficient and effective way [54]. The algorithm first computes empirical mutual information of all pairs of variables using their sample values. Then, it finds the maximum weight spanning tree with edge weights equal to the mutual information between the variables connected by the edge. We do not use any prior knowledge regarding the hierarchical dependency among object categories during the learning procedure. The Chow-Liu algorithm can simply select strong pairwise dependencies.

The aim of re-inference of using three relational models is to select the most evidently possible and contextually plausible object seeds that will be used for later

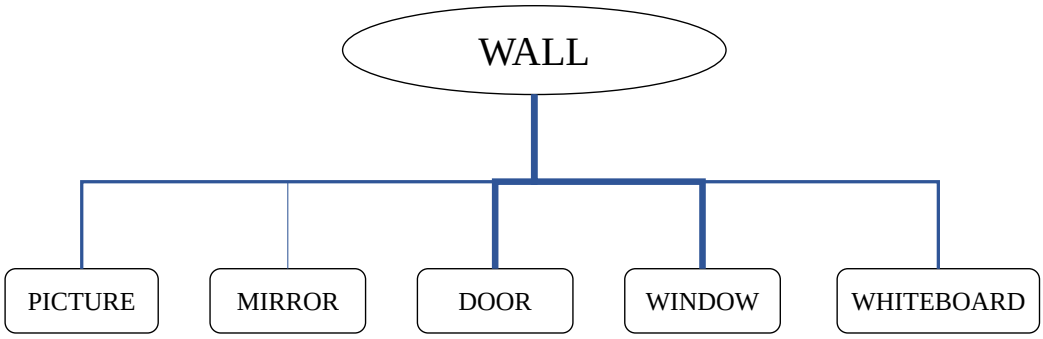

Figure 6.7: The NPM learned from the SUN RGB-D dataset [2] that includes all objects hung on the wall. The thickness of blue edges show the closeness to the wall, and non-hangable objects have been trimmed off during the training process.

optimization of bounding box or instance segmentation. The input of re-inference is the output (before the soft-max layer) of any deep networks for RGB-D semantic labeling, noted as the 3D score map L_0, which stores the scores for all category channels. The output of re-inference is a grid-based object seed set S. Then instance masks are generated for each element in the object seed set S. Specifically, the re-inference algorithm has three main steps: (1) Initialization of the 2D layout map; (2) Local seed initialization via VPM; (3) Contextual seed screening where HPM and NPM are involved. Step 3 is an iterative process that is intended to suppress unlikely objects and boost possible ones progressively.

Layout map initialization

Given the fact that most major indoor objects are placed on the ground, the indoor object placement can be inferred from a 2D layout map of top-down bird's eye view. To do so, we can discard the vertical axis of L_0 of which the dimension is $(3+C)\times M$,

to get the layout map L with dimension $(2+C) \times M$, where "2" indicates that L is a 2D layout map, N is the number of category channels, and M is the number of total 3D points. In the following re-inference, L will be updated with the help of three relational models from which the set of object seeds is obtained.

Local seed initialization

We make the first re-inference on L with the VPM R_V by focusing only major base objects. Given the VPM R_V, we update each major object category channel L^c in the layout map L in a point-by-point fashion as:

$$L^c = \max_{d_c : R_V(d_c, c) > 0} \{L^{d_c}, L^c\}, \tag{6.8}$$

where each d_c is a minor object category with strong co-existence with the target major object c in VPM R_V. This aims to take advantage of the detection of minor objects to support the related major objects along the vertical direction. For instance, the detection of a pillow reinforces the detection of the bed underneath. The updated layout map L will used to create the set of object seeds. We use a grid-based method [46] to regulate the object density and reduce the computational load. We divide L equally into $f \times f$ cells. Then we generate the grid-wise point set $Q = \{q_{i,j} | i, j = 1, 2, ..., f\}$ by finding the local representative point set in each cell$q_{i,j}$ as follows:

$$q_{i,j} = \arg\max_{c=1,...,C} L^c(p), p \in l_{i,j} \tag{6.9}$$

where p represents each point in a cell $l_{i,j}$ and $L^c(p)$ is a category-specific score of channel c for point p in the scope map L. It is worth mentioning that the number of points p in each cell is not the same due to non-uniform density during the 3D-2D projection from L_0 to L. However, the points along the vertical direction always belong to the same object. Thus, this process does not cause under-detection but facilitates the detection of base objects that is expected to benefit the detection of surrounding objects via HPM and NPM. The scene-level object seed set S is initialized

by the local winning set Q both of which will be updated iteratively in the following step.

Contextual seed screening

Given the current seed set S, we can gradually update the score map L with respect to other object categories by considering contextual information encoded in HPM R_H and NPM R_N. This step aims to enhance the score of minor objects that are not in S yet but have co-existence relationship with those in S or strong dependency with the detected wall. This is done in a point-by-point fashion for the score maps of object categories absent in S as follows:

$$\begin{aligned} L^c &= \gamma \cdot L^c, \\ \text{s.t.}\{R_H(c,s) \neq 0 \ \ \text{or} \ \ R_N(c,W) &\neq 0, s \in S, c \notin S\}, \end{aligned} \tag{6.10}$$

where γ is the scaling factor (> 1) to boost the scores of objects c in the layout map L due to its strong co-existence with other objects in S. After L is updated, the local winning set Q is recalculated that is further used to update the scene-level seed set S via collective pair-wise co-existence scores:

$$S = \arg\max_{S \subset Q} \{ \sum_{(s_i, s_j) \in S} R_H(s_i, s_j) \}, \tag{6.11}$$

where s_i and s_j are any current object pairs in S. As the result, a pair of co-existent objects will be preserved, and objects that are exclusive with those with stronger scores will be rejected. The updated S is not only more evidently likely (according the score L) but also more contextually plausible (according to HPM).

The whole re-inference algorithm works as a variation of alternative inference: (1) Firstly, we condition on the local winning set Q to infer the object seed set S. (2) Then we use the object seed set S, which represents the objects so-far-found in the scene, to update the layout map L and get an updated Q. Note that only the

Algorithm 1: Re-Inference algorithm

Convert the 3D score map L_0 from a trained network to a 2D layout score map L that has C category-specific channels;
Update each channel L_c in L by applying VPM R_V using the (8);
Divide L equally into $f \times f$ cells following the method in [38];
Generate the winning point in each cell of L to get the local winning point set Q using (9);
Initialize the object seed set S by finding the highest-scored-point in Q;
for $i = 1$ *to* N **do**
    Update L by applying HPM R_H and NPM R_N given the seed map S using (10);
    Re-generate Q from L using (9);
    Find the compatible new set S from Q given R_H using (11);
end
Output: the object seed set S; updated layout map L

object seed set S is used in the next step to generate bounding boxes. The complete re-inference algorithm is detailed in the pseudo-code.

6.3 Instance segmentation

The basic idea of our bounding box generation algorithm is to fully use the category-specific score-map L. After finding the object seed set S, we generate the instance masks for each element in L using the updated score-map L in the form of a 2D bounding box. To get the bounding boxes in 3D space, we will retrieve the object height by finding the classified points in the original 3D score map L_0. In this section, we first discuss a baseline method for bounding box generation based on pixel-level labeling results; then we introduce our objective function for instance mask evaluation that involves confidence scores and depth visibility; lastly, we present an efficient optimization algorithm that involves region detection and local fine-tuning.

6.3.1 Instance mask initialization

Our objective is to generate the holistic bounding boxes from the detailed pixel-level labels. Thus, we use the object seed set S and corresponding updated layout score map L to generate the bounding box, where S indicates which objects exist in the scene with a rough location as single seed points; L is boosted for all the relevant

objects to ensure the detection of the full shape of them. We generate one bounding box for each seed in S, optimize each bounding box, then eliminate the duplicates as the last step. To initialize the instance masks which are in the form of bounding boxes, we assume the seed point s_i to be in the center of the bounding box because the center part is considered to be supported by the surroundings which could lead to a high score in the score maps. The bounding box dimension is initialized using the statistical values of the target class. Since the bounding boxes are generated from the 2D layout score map S, only one dimension rotation is considered and randomly initialized. Since the score-maps from the deep learning network have relatively low resolution, the rotation angle r is optimized with a 5-degree step-size in this work.

6.3.2 Instance mask optimization

For each box seed s_j, we define the corresponding B_j as the bounding box parameter set. The bounding box we generate should agree with both the deep-learning inference results and the visibility in the scene. To ensure the generated bounding box does not occlude the visible items in the scene, we remove the "floor" points, quantize the layout map into grids and label all the empty ones to get the empty space map M. Thus, for a bounding box candidate B_j, the evaluation function is defined as:

$$E_i^k = \alpha * F(B_j, L) - \beta * G(B_j, M), \tag{6.12}$$

where $F(\cdot)$ returns the average score of B_j from the layout score map L and $G(\cdot)$ returns the visibility penalty when B_j conflicts with the empty area M, as defined below:

$$G(B_j, M) = \frac{H(B_j, M)}{V(B_j)}, \tag{6.13}$$

where $H(\cdot)$ finds the amount of empty grids M in bounding box hypothesis B_i^k; and $V(\cdot)$ returns the 3D volume of a bounding box. This indicates that for small objects, our method has relatively small tolerance to ensure the majority of the object volume

to be solid. During the optimization process, E_i^k will be maximized by adjusting bounding box parameters in B_i^k. α and β are weights to balance the two terms in (6.12). Our optimization process has two steps: global rough search and the local fine-tuning. In the first step, to capture all possible bounding boxes in the scene, for each seed point s_i in a grid cell $cell_j$, the bounding box is initialized using the object mean size. During optimization, the seed point s_j must always stay in the bounding box to preserve the diversity in 3D space and reduce the computational load. In the second step, we optimize the 1D rotation along the gravity, the boundaries of each of the four edges in top-down view. That is a total of 5 parameters. The optimizer we use is a modified heuristic direct search method [59]. The direct search optimizers are simpler and faster than gradient-based ones. We iteratively optimize the 5 parameters of each bounding box. Instead of using a fixed step size during optimization, the search rate r is updated adaptively for each parameter as follows:

$$r_{new} = \frac{E_{new} - E_{old}}{|E_{new} - E_{old}|} * t_1 * r_{old} + t_2 * r_{old}, \tag{6.14}$$

where E_{new} and E_{old} are the evaluation score after and before a step move, and t_1 and t_2 where $t_1 + t_2 = 1$ are used to balance the intended step change and the original step size, respectively. In our experiment, we use $t_1 = 0.75$ and $t_2 = 0.25$. We use physical constraints to find the height of the bounding box. If the bounding box shows a major object, it must stand on the floor. Otherwise, for a minor object, it must have a major object underneath.

6.4 Experimental results

We conduct the experiments on the SUN RGB-D dataset [2] and the ScanNet dataset [3]. In the following, we will first discuss baseline generation followed by detailed results on two datasets.

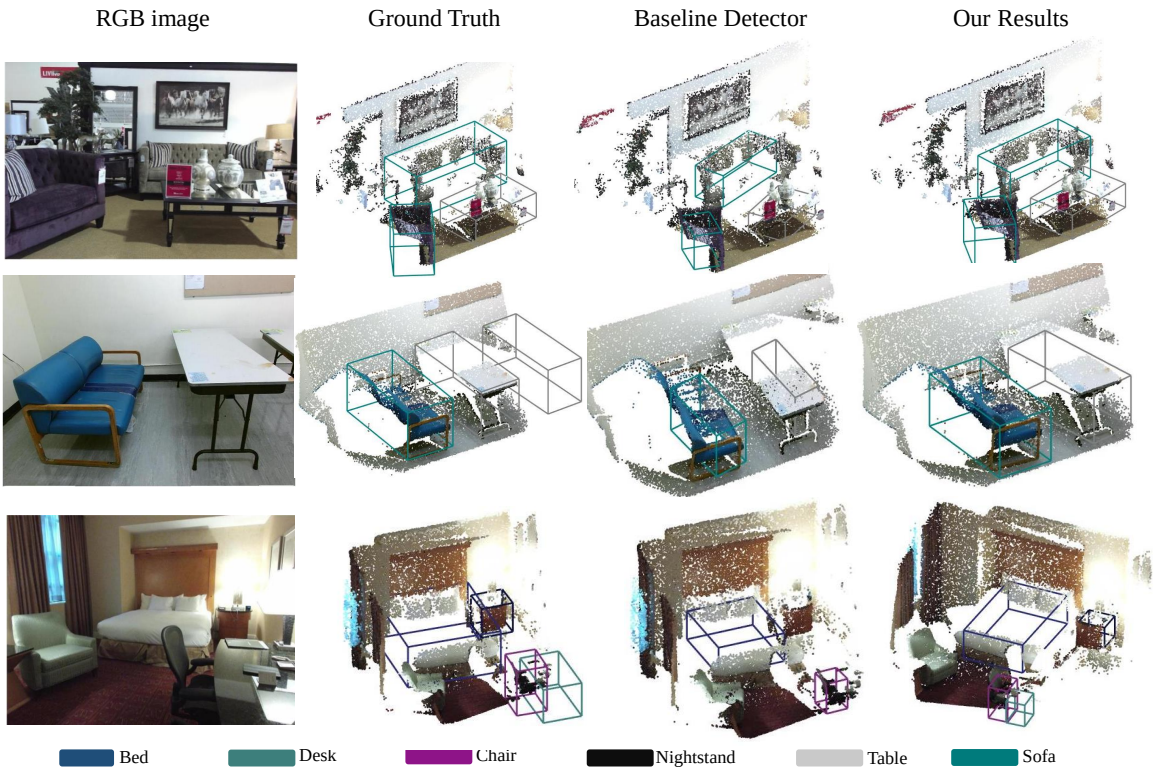

Figure 6.8: Some qualitative results for dataset SUN RGB-D [2]. Our method generates more accurate bounding boxes than the baseline detector [1]. Moreover, the example in the third row shows our re-inference approach brings back the nightstand and the desk. However, our method cannot estimate the full shape of the object if it is out of the field of view or even fail if only a small part is captured in the image (row 2).

6.4.1 Baseline generation

Given a pixel-level segmentation map, we transfer the class label to the RGB-depth data points and then we generate L from the labeled 3D points in the same way we created training data for VPM and HPM. For a specific given object, the size of the bounding box is obtained from the range of data points labeled as that object. Thus, we need only to find the bounding box orientation. We simply generate the bounding boxes for all directions with a step size of 5 degrees. Then, the bounding box with least points from other categories is considered to be the tightest and is selected as our initial bounding box. Given the layout view points set L, the 2D bounding box $\hat{X}_i$ is generated as:

$$\hat{X}_i = \arg\min_{r \in \{0,\pi\}} \{E_0(X_i(r))\}, \tag{6.15}$$

where $X_i(r)$ is the bounding box parameter set for object i with orientation r in the layout view L. The bounding box evaluation function E_0 gets the total point number that falls within $X_i(r)$ but not classified as object category i in the layout view L. To get the 3D bounding box, we add the height to X_i using the highest (along the gravity) labeled data point in X_i. Here we consider three kinds of objects, base objects, accessory objects and individual objects as defined before. Although pixel level labeling provides more object categories, we only consider the objects with ground truth bounding boxes.

6.4.2 SUN RGB-D dataset results

The SUN RGB-D [2] dataset includes 5285 training RGB-D images and 5050 test images with ground truth bounding boxes. We follow the ground-truth given by the dataset to use 3D bounding boxes to present the instance segmentation. The camera parameters are given and the projected 3D points are aligned with gravity. We use a semantic labeling network that considers 37 categories [1] to generate 10-class bounding boxes following [7]. Due to the fact that the view range is limited

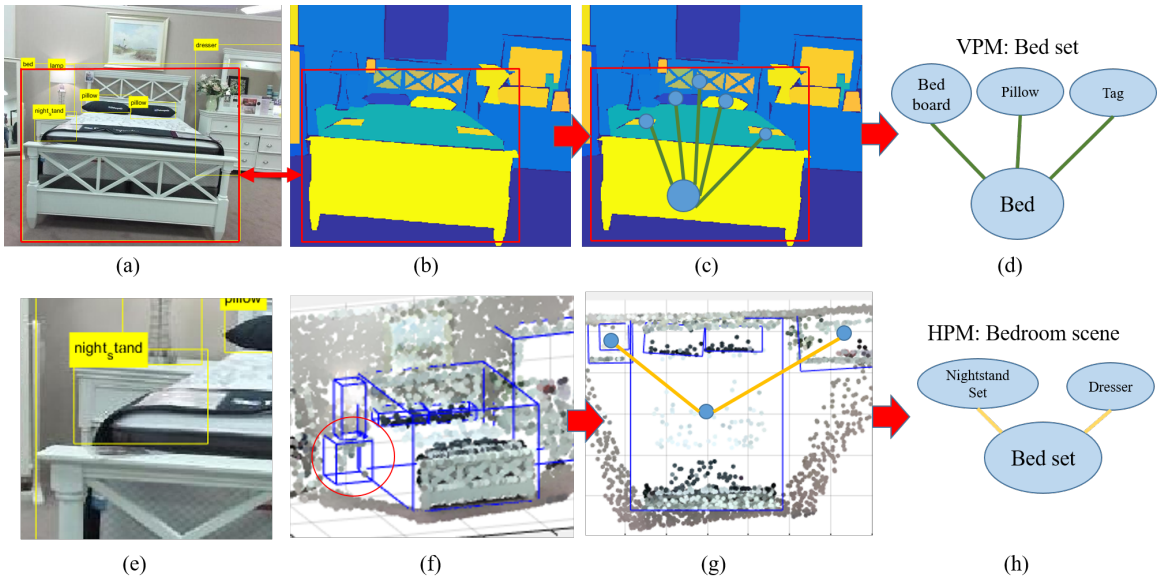

Figure 6.9: Illustration of VPM and HPM for SUN RGB-D dataset [2]. (a) A bedroom image. (b) Ground-truth pixel level labeling of (a) where the bed is labeled as multiple items. (c) The vertical placement relationship of the bed set. (d) VPM for the bed set. (e) The cropped portion of the nightstand in (a). (f) Ground-truth bounding boxes in 3D space. (g) The horizontal object placement from the top-down view. (h) HPM for the bedroom.

for a single RGB-D image, we cannot get high quality wall detection. Thus, the NPM model, which describes the relations between hangable objects and walls, can hardly be found. Moreover, no hangable object exists in the 10 target categories. So, we neglect the NPM in this dataset. The whole process is illustrated in Fig. 6.9. To generate the baseline bounding boxes, we use the pixel level labeling map from a deep learning method [1] as the input for the algorithm described in Section 4.1. The widely used evaluation measures mean average precision (mAP) under certain IoU threshold (0.5). Note that in indoor scenes with heavy occlusion and sparse 3D data points, some of the bounding box ground-truth are inferred from the visible area. Moreover, the bounding boxes are manually labeled by different people that are with inconsistent standards. Thus, the ground-truth bounding boxes are more like references rather than the true ranges of objects.

From the results listed in Table. 6.1, we can see that our method brings more benefits to the small objects where the deep learning detectors are less effective. The leading algorithm [19] gives very high detection accuracy on typical big objects which have plenty of details in the input images. Thus, the performance on these big objects are mainly from the strength of the deep-learning network detectors. While the small objects, on the other hand, could be boosted by context. We could find our method reaches a comparable level of the small objects compared to other state-of-art methods. Note that our method find the nightstand, which is hardly found by the baseline detector. The problem actually comes from inaccurate labeling of the nightstand, table and dresser. The confusion comes more from the nightstand and the end-table, which is sometimes labeled as table. With the context boost, our method could re-find the nightstand even though the detector is heavily contaminated by the inconsistent ground-truth. In the semantic labeling task, it considers many more categories. Our method is easy to extend to generate instance segmentations for other categories, while the traditional networks require retraining the whole network

	Comparisons			Ablation Study		
Category	$PointNet$ [19]	$VoteNet$ [62]	COG [63]	$Base$	$Base + V$	$Ours(Full)$
Bathtub	37.26	74.40	76.20	21.27	22.13	48.02
Bed	68.57	83.00	73.20	62.31	71.03	71.74
Bookshelf	37.69	28.80	32.90	28.10	34.29	40.26
Chair	55.09	75.30	60.50	56.79	56.79	65.07
Desk	17.16	22.00	34.50	16.41	19.23	28.00
Dresser	23.95	29.80	13.50	21.08	22.32	23.87
NightStand	32.33	62.20	30.40	0	11.16	28.67
Sofa	53.83	64.00	60.40	41.17	48.52	56.00
Table	31.03	47.30	55.40	37.78	42.45	51.29
Toilet	83.80	90.07	73.70	70.74	71.03	78.04
MEAN	44.07	57.69	51.07	33.86	39.90	49.10

Table 6.1: The evaluation results (%) of dataset SUN RGB-D [2] in terms of bounding box mAP 10 major indoor objects. $Base$ is our baseline algorithm flow; $Base + V$ is our method with the VPM only; $Ours(Full)$ is our full version algorithm that considers the two context models.

over again.

6.4.3 ScanNet dataset results

ScanNet is an RGB-D video dataset containing 2.5 million views in more than 1500 scans, annotated with 3D camera poses, surface reconstructions, and instance-level semantic segmentations. The advantage of ScanNet dataset is the vision of the whole scene. Unlike the single-view dataset SUN-RGBD, ScanNet enables all the objects visible in the layout. Although partial occlusion still exists due to the complexity of the indoor scenes, the dataset has well preserved the presence information of all objects. Thus, our context-based algorithm is well supported. We follow the ground-truth to use the point-level mask to present the instance level segmentation. Since the ScanNet dataset provides scene level 3D point clouds that are computationally high, we collapse the whole scene onto the ground to use the layout map. Moreover, since almost all the objects appear in the scene level, we limit the coexistence to be within the wall constraint. The walls and floor points are not considered during the mask generation process. Unlike the fixed grid size used in the SUN RGB-D dataset [2], we generate distance-determined grids for the seeding process. The grids are generated for each 0.5 meter range.

From the Table 6.2, we can see our approach improves the detection rate from the baseline method. Some qualitative results can be found in Fig. 6.11, which shows the effectiveness of our method in finding the object full shape, removing outliers and segmenting each single instance. Some more qualitative results for different object categories are shown in Fig. 6.12. In general, out method is very effective for those small objects that could be directed from a typical big object. Our baseline method is a semantic labeling network. It is possible that the baseline method cannot catch some of the object features, which limits the potential performance in instance segmentation. However, semantic segmentation provides us valuable information of

	Comparisons			Ablation Study			
Category	$MTML$ [64]	$BoNet$ [65]	$PANO$ [66]	$Base$	LO	$V+H$	$Ours(Full)$
Bathtub	100.0	100.0	66.7	31.0	48.3	78.1	78.1
Bed	80.7	67.2	71.2	62.2	71.8	73.3	73.3
Bookshelf	58.8	59.0	59.5	53.2	53.2	55.5	55.5
Cabinet	32.7	30.1	25.9	21.1	36.6	38.1	38.1
Chair	64.7	48.4	55.0	17.6	35.4	51.0	52.5
Counter	0.4	9.8	0.0	11.5	14.3	14.3	14.3
Curtain	81.5	62.0	61.3	21.5	21.3	51.7	74.6
Desk	18.0	30.6	17.5	21.1	35.8	34.3	34.3
Door	41.8	34.1	25.0	8.6	18.8	26.9	38.1
OtherFurniture	36.4	25.9	43.4	23.4	34.5	35.2	35.5
Picture	18.2	12.5	43.7	18.9	26.2	23.1	25.7
Fridge	44.5	43.4	41.1	12.6	38.1	42.2	42.2
ShowerCurtain	100.0	79.6	85.7	43.8	55.9	68.1	76.8
Sink	44.2	40.2	48.5	15.0	28.1	40.2	40.2
Sofa	68.8	49.9	59.1	51.5	57.3	62.0	62.0
Table	57.1	51.3	26.7	35.5	35.8	42.2	42.2
Toilet	100	90.9	94.4	86.6	85.3	87.8	87.8
Window	39.6	43.9	35.9	9.3	10.8	27.6	36.0
Average	54.9	48.8	47.8	30.2	39.3	47.1	50.2

Table 6.2: The evaluation results (%) in terms of mAP(mean Average Precision) using the IoU (Intersection over Union) threshold as 0.50. 18 indoor objects are considered following the ScanNet dataset benchmark [3]. $Base$ is the baseline algorithm flow; LO is our method with only local optimization; $V+H$ represents our algorithm with two context models (VPM and HPM) along with local optimization; $Ours(Full)$ is our full version algorithm by adding the NPM.

Figure 6.10: Some scene scan examples in the ScanNet dataset [3].

some classes other than those in the instance segmentation benchmark. For example, with the information of the class of *wall*, Objects like *door*, *picture*, *window* and *curtain* gain significant improvement, which is mainly from the NPM for their spacial connection to the walls.

6.5 Discussion

We have presented context-based graphical models for relational modelling of indoor object categories, i.e., the vertical placement model (VPM), the horizontal placement model (HPM) and the non-placement model (NPM). Specifically, the VPM captures the co-existence of major and accessory objects; the HPM encodes ground level spatial configuration of different individual objects; and the NPM describes the instances that are hung in the scenes rather than placed with the reference to the gravity. The context model allows us to bridge the gap among the three levels of semantic scene understanding: from pixel-level labeling to instance-level identification to scene-level object grouping. As a case study, we apply the context-based models in a bottom-up

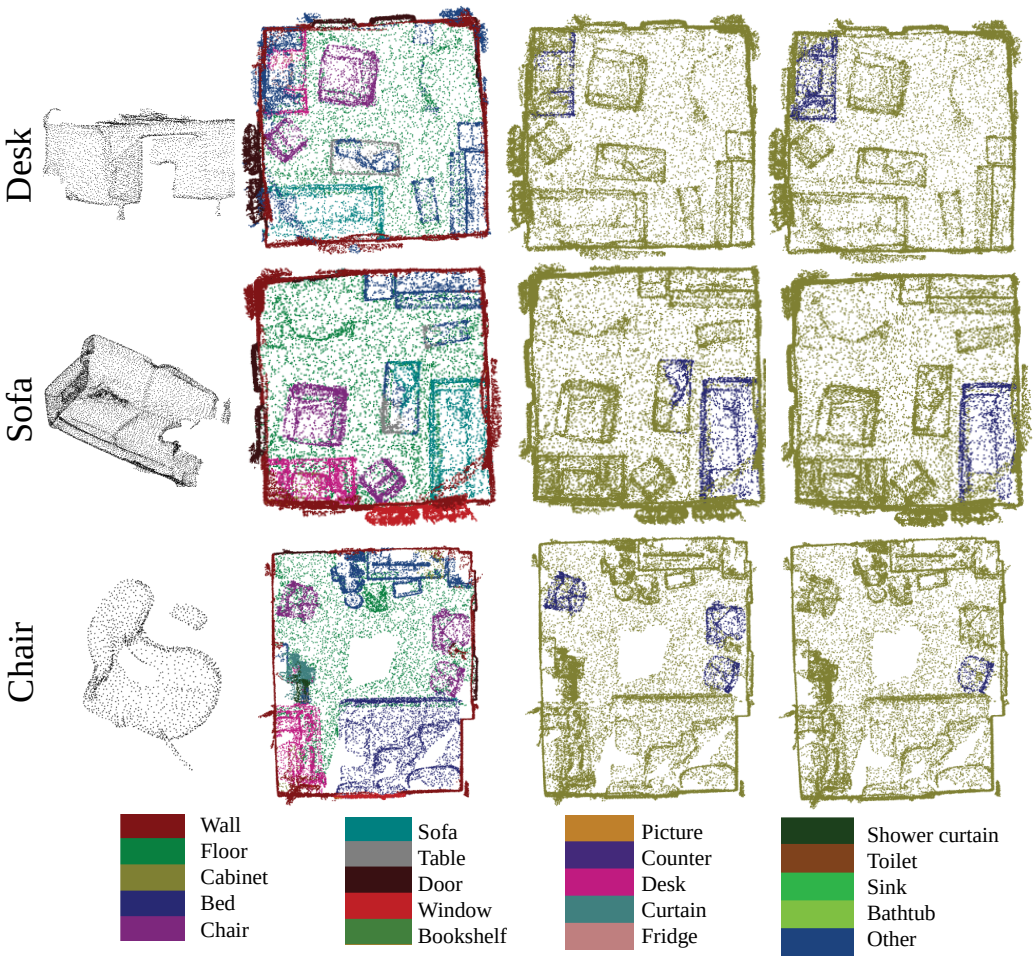

Figure 6.11: Some qualitative results for the ScanNet dataset [3]. The columns from left to right are: target object ground-truth segmentation; detector output of the whole scene; target object directly from the detector output; final object segmentation after our approach. With our approach, the boundary of the desk is better found; the outlier of the sofa is removed; the chairs in the scene can be separated as single instances.

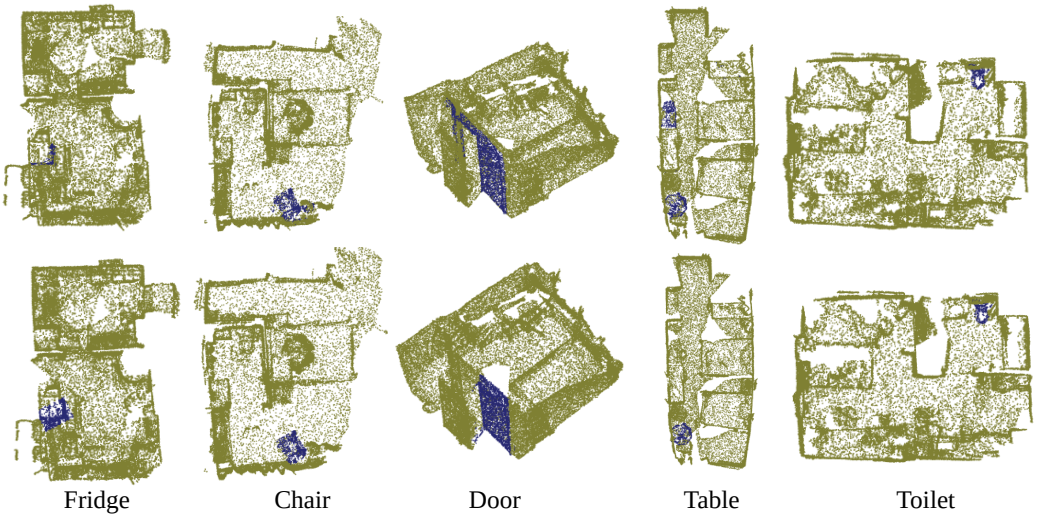

Figure 6.12: Some more random examples in the ScanNet dataset [3]:(1) the detection of a featureless fridge is hard for detectors but gets boosted by our relational model; (2) instance masks become more accurate by removing classification outliers as the 2^{nd} and the 3^{rd} columns show; (3) multiple instances of the same category can be separated, as shown in the 4^{th} column where two tables are segmented; (4) for the easy-to-detect objects with typical appearances as the toilet in the 5^{th} column, the detector is able to make accurate detection that leaves little space to improve.

flow to create object-specific instance-level segmentation in 3D space that is more informative and intuitive where the input is the pixel-level label result from any deep neural network. Experimental results show the promise of our context model to improve the quality of instance segmentation. It is foreseeable the our method can be used in other holistic scene understanding tasks.

CHAPTER VII

LOCAL COLLABORATIVE OBJECT PRESENCE NETWORK

In this chapter, we use the the context information to boost indoor scene semantic segmentation using deep learning networks. A re-inference module is introduced to reinforce the semantic labeling baseline network by introducing the Object Collaborative Presence(COP) as the context information. The method is upgraded as Local COP(LoCOP) to further improve the segmentation accuracy. Our re-inference network architecture can be applied to any deep learning frames and be easily guided with specific high-level prior knowledge.

In this chapter, we have the following sections: (1) the preliminary work about encoding context information to networks; (2) Collaborative Object Presence(COP) network (3) Local Collaborative Object Presence(LoCOP) network; (4) Experimental results; (5)Discussion.

7.1 Preliminary work

The deep learning technique develops fast in these years. With different kinds of network architectures, it has potential to solve a variety of real word problems [67,68]. Different from the conventional end-to-end tasks for deep learning networks, recent researchers try to transfer knowledge to the networks [10,69,70] to improve robustness, to speed up the training process or to obtain better performances, as shown in Fig. 7.1. To fully utilize the low-level features from a well-trained network, researchers widely use the backbone technique which is to initial the low-level network layer weights using the well-trained network with a similar task [71–73]. However, the backbone method

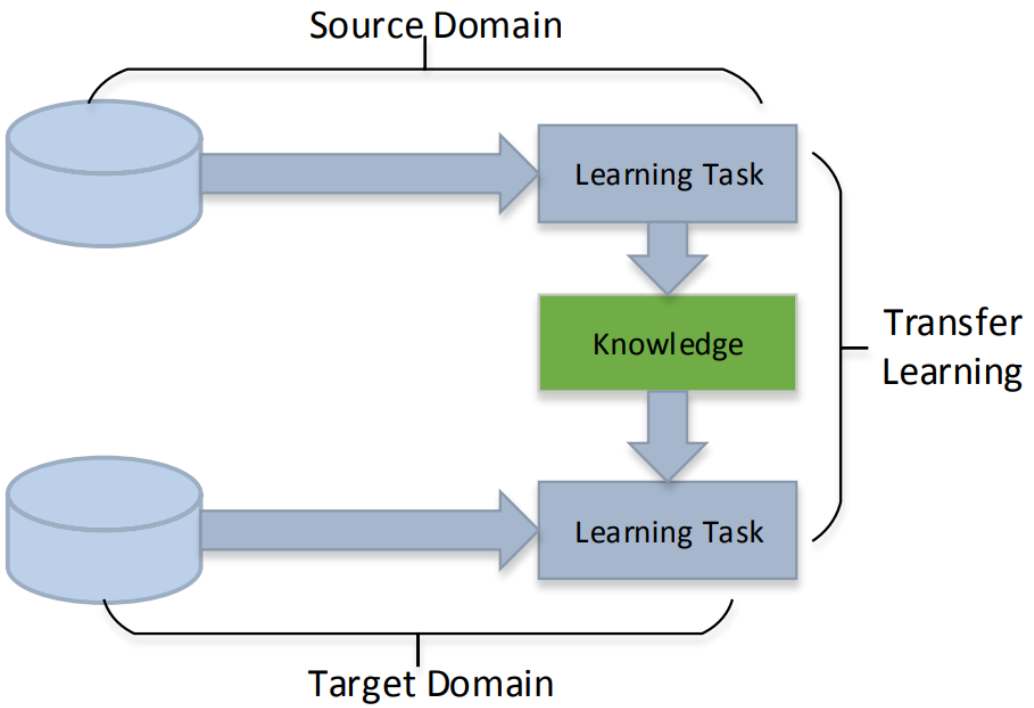

Figure 7.1: The idea of transfer learning [10]. The knowledge obtained from the source domain is applied to guide the learning task in the target domain.

works only for the low-level features. It is very hard to implement any high-level understanding that people posses.

To use people's prior knowledge in indoor scene understanding, researchers have proposed method to utilize context information to guide the networks in semantic segmentation tasks [11, 74, 75]. Semantic segmentation assigns per-pixel predictions of object categories, which provides a comprehensive scene understanding including the information of object category, location and shape. Current semantic segmentation approaches are typically based on the end-to-end framework [6, 36, 76]. As the network structure shows in Fig. 7.2, the method proposed in [11] finds the context features in the context encoding module, fuse them as global regulations in the final segmentation stage. Given an input image, a pre-trained CNN is used to extract dense

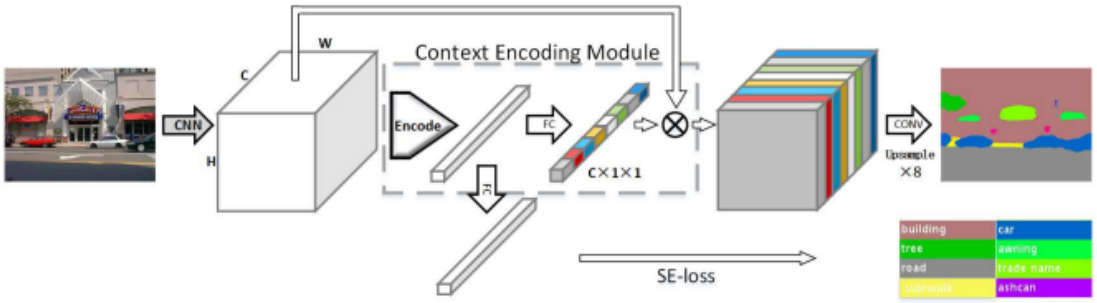

Figure 7.2: Overview of the the preliminary research EncNet [11]. A Context Encoding Module with an Encoding Layer to capture the encoded semantics. The Semantic Encoding Loss (SE-loss) is calculated to regularize the training which lets the Context Encoding Module predict the presence of the categories in the scene. Then the prescenced classes are highlighted before they are fed into the last convolutional layer to make per-pixel prediction.

convolutional feature maps. A Context Encoding Module is built on top, including an Encoding Layer to capture the encoded semantics and predict scaling factors that are conditional on these encoded semantics. These learned factors selectively highlight class-dependent feature maps (visualized in colors). In another branch, Semantic Encoding Loss (SE-loss) is calculated to regularize the training which lets the Context Encoding Module predict the presence of the categories in the scene. Finally, the representation of Context Encoding Module is fed into the last convolutional layer to make per-pixel prediction.

As shown in Fig. 7.2, their contextual information is implemented by adding a weight ratio for the whole image channel before the final convolutional layers for labeling. However, the contextual information is impaired since the weights of the upcoming convolutional layers could easily mitigate the contextual weights. Moreover, it is unwise to suppress the whole channel for the whole scene because the detector process and evaluate each pixel respectively. It is highly possible that the detector

works well in some parts of the image but does a poor job in some other parts. Our work is inspired by their network structure and we focus on the extraction method and accuracy of the context feature itself without worrying about the final instance segmentation. We propose the Collaborative Object Presence(COP) feature and its upgraded version Local Collaborative Object Presence(LoCOP) feature.

7.2 Collaborative object presence (COP) network

To use the scene level information as context to guide the networks in semantic labeling, we propose the Collaborative Object Presence(COP) feature. which is a binary vector that indicates which objects presence in the scene. The information is used as an surveillance in the labeling modules that come next: to remove the incompatible objects and to enhance the detection of the presence object.

The architecture of our network is described in Fig. 7.3. The upper branch gets the scene-level COP vector. To apply COP into the final labeling module, we extend each bit in the COP vector to have the same dimension as the scene score maps, concatenate them and feed to the final labeling layers. Thus, each kernel in the coming convolutional layer has the COP information for every part of the image.

In standard training process of semantic segmentation, the network is learned from isolated pixels (per-pixel cross-entropy loss for given input image and ground truth labels). The network may have difficulty understanding context without global information. To regularize the training of COP channels, we calculate loss on the COP vector which forces the network to understand the global semantic information with very small extra computation cost. We build additional convolutional layers with a sigmoid activation function on top of the concatenated score maps to make individual predictions for the presences of object categories in the scene.The COP detection is trained separately with binary cross entropy loss that considers big and small objects equally. Thus, in the next labeling module, the segmentation accuracy

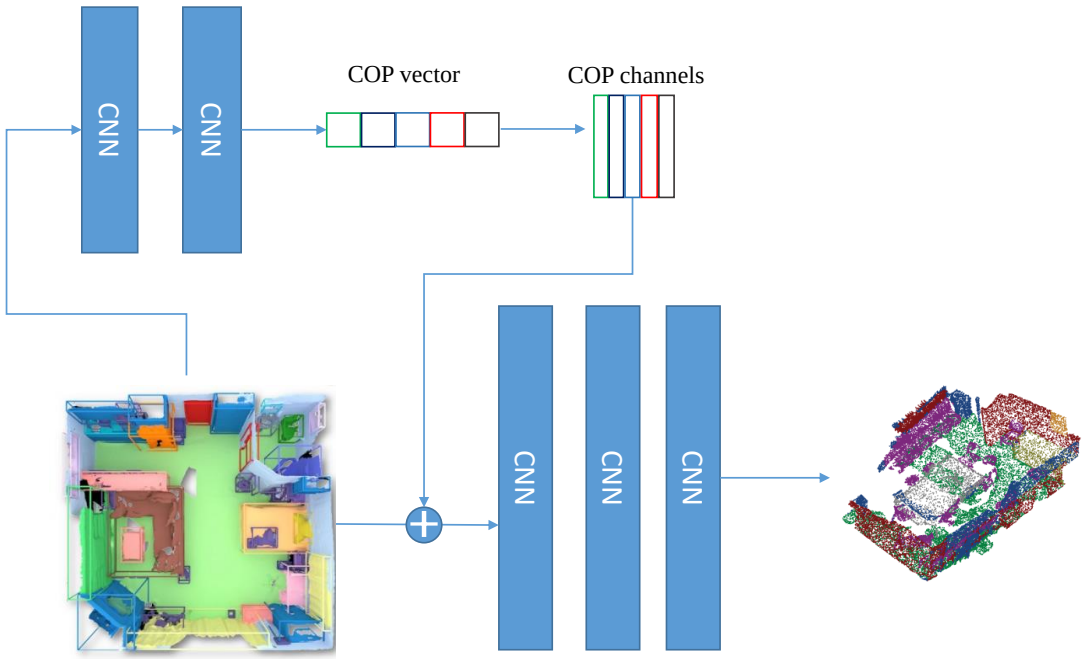

Figure 7.3: The COP method: start from the scene score maps, the upper branch generates the COP feature

of small objects are often improved. To extract the COP vector, we use 2 layers of Convolutional Neural Networks (CNN) with 256 hidden neurons followed by a Fully Connected Network (FCN) layer. The final labeling module consists of 3 layers of CNNs and one soft-max layer to get the final output.

The disadvantage of COP is that the information it provides is for the scene level, which could be too vague for local labeling inference. We expect the COP that could carry the scene-level compound knowledge, but still could be helpful for local tasks since the inaccuracies happen around object boundaries. This leads us to the next section: Local Collaborative Object Presence(LoCOP) network.

7.3 Local collaborative object presence (LoCOP) network

With the proposed COP structure described in the previous section, we build a Local COP network to enhance the local guidance from the contextual information. We follow the method in previous chapters to divide the scene into grids then each grid contributes one bit in the LoCOP vector. Different from the binary COP, LoCOP has integers as its value to represent each categories. As shown in Fig. 7.4, the full LoCOP is a map and get divided into the same patches as the input scene. Each patch has $2x2$ grids that are used during the re-inference process.

The full architecture of LoCOP is shown in Fig. 7.5. The LoCOP map is trained individually on the scene level information to predict the presence of the object class in each grid. We use 2 layers of Convolutional Neural Networks (CNN) with 256 hidden neurons. When doing the re-inference, the map is got divided according to the scene patches. Similar as the COP structure, the LoCOP channels are expanded to the same dimension as the score maps of the patch, which means that each channel contains one label information for the specific position. Similar as the COP architecture, the final labeling module consists of 3 layers of CNNs and one soft-max layer to get the final output. After the re-inference process, the patches are put back together to be

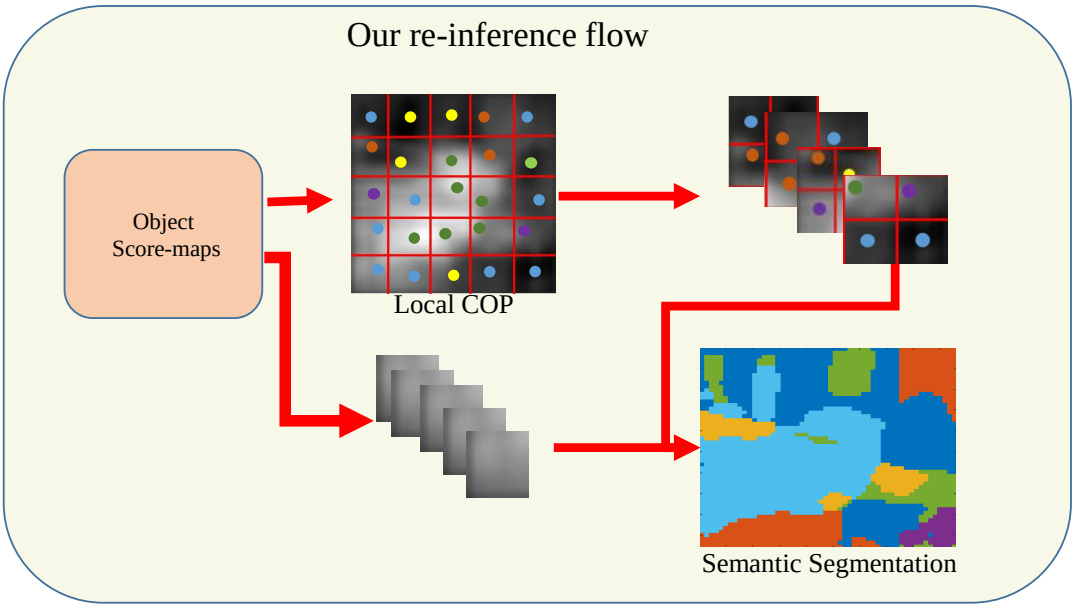

Figure 7.4: The LoCOP method obtains the COP feature for the whole map. Then the inference is done for the scene patches.

the final scene level result.

The ground truths for LoCOP map are directly generated from the ground-truth segmentation mask without any additional annotations. Our COP and LoCOP module is differentiable and inserted well with the existing semantic segmentation pipelines without any extra training supervision or artificial prior knowledges. In terms of computation, both COP and LoCOP are light-weighted since the modules aim only for re-inferencing.

7.4 Experimental results

We conduct the experiments on the ScanNet dataset [3]. ScanNet is an RGB-D video dataset. It has more than 1500 scene-level scans that are generated from 2.5 million views that annotated with 3D camera poses, surface reconstructions, and instance-level semantic segmentations. The advantage of ScanNet dataset is the vision of the

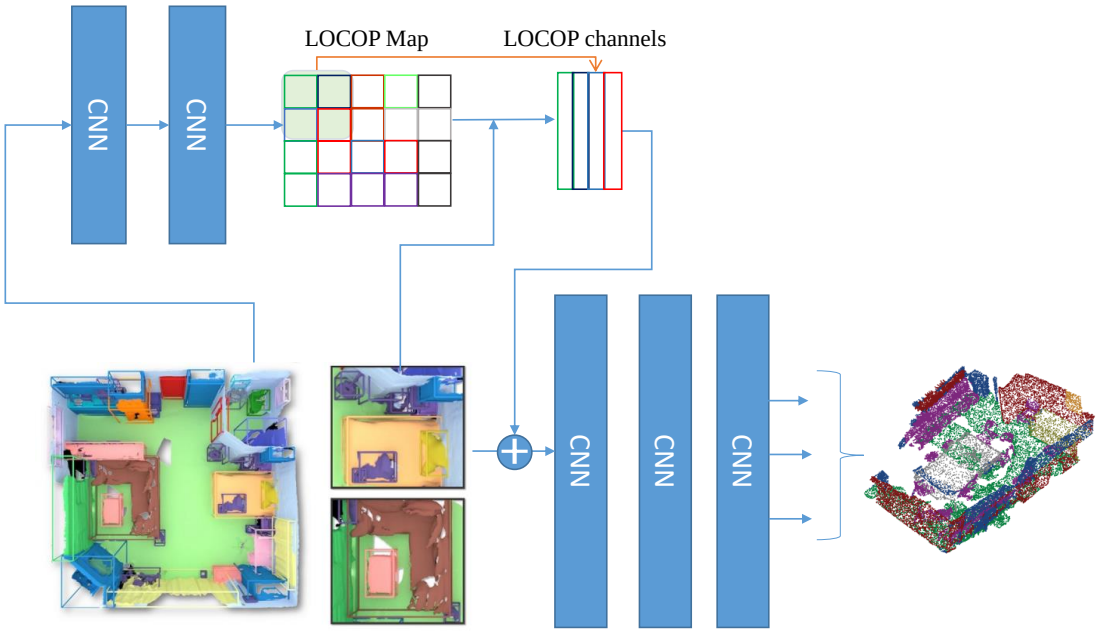

Figure 7.5: The architecture of our LoCOP network: the LoCOP map is trained for the scene level and divided into patches for inferencing to provide top-down regulation for the target local area. The patches are put back together to obtain the final output. Since the patches are directed by the same LoCOP map, the divide-and-merge process does not increase the labeling difficulty.

whole scene. Unlike the single-view dataset SUN-RGBD with occlusion and limited field of view, ScanNet enables all the objects visible in the layout. Although partial occlusion still exists due to the complexity of the indoor scenes, the dataset has well preserved the presence information of all objects. Note that different from the previous chapters, semantic labeling is conducted in this chapter and the evaluation is the labeling accuracy across each pixel.

Since the ScanNet dataset provides scene level 3D point cloud which is computationally high. We project the whole scene onto the ground to use the layout map as we described in the previous chapters. We generate distance-determined grids for the LoCOP map. The grids are generated for each 0.2 meter range. The local patches are divided for each $2x2$ grids. Our experiment system is built in PyTorch. We use the base learning rate as 0.01 and weight decay rate is set to 0.9. The networks are trained for 50 epochs for COP/LoCOP feature extraction and 120 epochs for the final classification module. We didn't use any data augmentation technique because they may impair the context information. For example, the widely used cropping method cuts the scenes into pieces which also cuts off the objects' possible relations. The rotation method works for shape-based features, but our COP/LoCOP is presence-based features.

From the results listed in the table, we can see that our method brings benefits to the objects in general and LoCOP helps more for the small objects where the deep learning detectors are less effective. The big and typical objects are normally with little occlusion and placed isolated, which makes them easy to be identified with clear boundaries, like the *bed*, $sofa$ and *toilet*. While the small and occluded objects, on the other hand, could be boosted by context. COP does limited improvement because the context information is very vague: the presence is for the whole scene. The presence information could hardly help to find the boundaries of objects. LoCOP not only provides local presence but also has orientational information since the object vector

	Comparisons		Our Method	
Category	SparseConvNet [12]	MinkowskiNet [77]	*COP*	*LoCOP*
Bathtub	64.7	85.9	65.2	70.1
Bed	82.1	81.8	82.1	82.1
Bookshelf	84.6	83.2	84.6	84.6
Cabinet	72.1	70.9	73.7	74.3
Chair	86.9	84.0	86.9	89.7
Counter	53.3	52.1	53.7	55.1
Curtain	75.4	85.3	75.6	76.8
Desk	60.3	66.0	64.4	67.4
Door	61.4	64.3	61.8	64.3
Floor	95.5	95.1	95.5	95.5
OtherFurniture	57.2	54.4	58.3	60.3
Picture	32.5	28.6	33.8	34.7
Fridge	71.0	73.1	71.3	75.1
ShowerCurtain	87.0	89.3	87.0	88.6
Sink	72.4	67.5	73.1	79.1
Sofa	82.3	77.2	82.3	82.3
Table	62.8	68.3	65.1	66.9
Toilet	93.4	87.4	93.4	93.4
Wall	86.5	85.2	86.6	87.0
Window	68.3	72.7	68.8	69.7
Average	72.5	73.6	73.1	74.8

Table 7.1: The evaluation results (%) in terms of average IoU (Intersection over Uion) for each catergory. 18 indoor objects are considered following the ScanNet dataset benchmark [3].

has a specific order. Thus, it does much more help to the non-isolated objects. We could find our method reaches competitive level of labeling objects compared to other state-of-art methods.

7.5 Discussion

We have presented a re-inference network module for indoor scene semantic segmentation using the context feature Local Collaborative Object Presence(LoCOP). Specifically, the LoCOP is a map that captures the co-existence indoor objects roughly without worrying about the real object boundaries. Then the LoCOP works as a prior knowledge to support the semantic segmentation which is processed in small patches to enhance the local detection. Experimental results show the promise of our LoCOP network to improve the quality of semantic segmentation. Similar as the widely accepted network backbone technique, the re-inference framework we propose can be applied to other holistic scene understanding tasks using any specific high-level knowledge. It shows great potential to fuse the bottom-up networks with the top-down human knowledge.

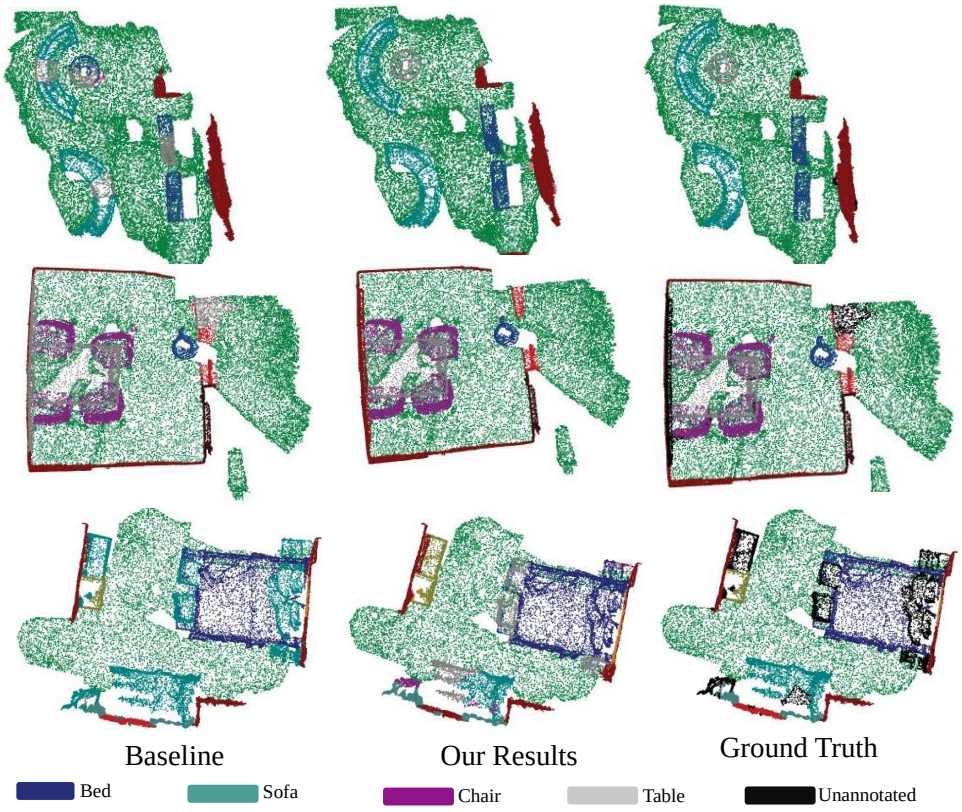

Figure 7.6: Some comparisons of qualitative results: the baseline method is [12]. Our method is able to correct some mislabeled points from the *Baseline*: the sofa and the table in the 1st row; the table in the 2nd row; the objects around the bed in the 3rd row. Note that the black points in the *Ground Truth* are *unannotated* points, which means that they are not labeled and will not be evaluated.

CHAPTER VIII

CONCLUSION AND FUTURE WORK

In this thesis, we focus on applying high-level context information to support object segmentation for indoor scenes. Our goal is to have an holistic scene understanding that is able to support applications such as robotics and navigation. To achieve this goal, we proposed approaches at four different levels of details. We start from the single instance representation level to context-supported instance detection level and finally to end-to-end instance segmentation. At single instance representation level, in chapter 3, we estimate the 3D room in cuboids to expose the local geometry of objects. Then in chapter 4, the idea is developed to find the instance level tight bounding boxes. Next, at the context-supported instance detection level, we investigate in the object co-existence patterns using modified graphical models. Then, we benefit from those algorithms and propose an end-to-end instance segmentation algorithm. At last, we further develop our idea to be part of the fast-developing neural network architectures. By adding a re-inference module with an independent pathway to find the object context information and then getting it back to guide the local detection results, our approach fuse the high-level knowledge to the network-based local detectors. We demonstrate the efficacy of each proposed approach with extensive experiments both quantitatively and qualitatively on public datasets.

8.1 Conclusion

The goal of this dissertation is to build computer systems that can see and understand our physical world in a way that they are able to safely interact with this world and

assist us in our daily lives. We believe that this requires machines to understand the complete 3D scenes around them, especially the instance relations in the scene, which is a task far beyond just labeling each pixel.

This dissertation work demonstrates that starting from the bottom-up pixel labeling detectors, the 3D representations and the context information are able to work jointly to outperform data-driven algorithms which are designed only for a specific instance segmentation task. Moving forward, I think it is also important to rethink the role of the benchmark tasks defined by open-access datasets. Rather than treating them as fixed hard limitations, which require researchers to get good scores in the table of a particular task, we should consider them as guidance of our 3D world, and make use of this to improve the true understanding of the environment holistically. By reconnecting task-specific detectors with high-level perception, we will be able to create more powerful and intelligent AI systems.

8.2 Future work

Indoor scene understanding is an comprehensive work that consists many parts including semantic segmentation, object detection, instance segmentation and so on. The state-of-the-art methods normally focus on one of these tasks. As shown in this dissertation, the approaches for the tasks could work jointly to create holistic scene understanding. Thus, our future work will aim on the following two points:

Prior knowledge implementation:High-level prior knowledge show its potential to guide data-driven approaches, like deep-learning networks, to acquire more comprehensive and accurate understanding. We investigate in the object presence in this dissertation, we will find and implement other context information to acquire more complete scene understanding in the future work.

Network architecture optimization: The network architecture we propose to implement the context information is trained individually from the local detectors.

The performance could be further improved if they are trained as a whole where the detector and the context information could support each other during the training process.

REFERENCES

[1] Z. Li, Y. Gan, X. Liang, Y. Yu, H. Cheng, and L. Lin, "LSTM-CF: Unifying context modeling and fusion with LSTMs for RGB-D scene labeling," in *Proc. ECCV*, 2016.

[2] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. CVPR*, 2015.

[3] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[4] *prototype robot for assisted living:*, 2009 (accessed February 3, 2014).

[5] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5556–5565, 2015.

[6] C. Couprie, C. Farabet, L. Najman, and Y. LeCun, "Indoor semantic segmentation using depth information," *arXiv preprint arXiv:1301.3572*, 2013.

[7] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Psongroc. ECCV*, Springer, 2012.

[8] H. Jiang and J. Xiao, "A linear approach to matching cuboids in RGBD images," in *Proc. CVPR*, 2013.

[9] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[10] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *International conference on artificial neural networks*, pp. 270–279, Springer, 2018.

[11] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context encoding for semantic segmentation," in *Proc. CVPR*, pp. 7151–7160, 2018.

[12] B. Graham, M. Engelcke, and L. van der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9224–9232, 2018.

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[15] B. Ayers and M. Boutell, "Home interior classification using sift keypoint histograms," in *Proc. CVPR*, 2007.

[16] L. Lingard, S. Espin, S. Whyte, G. Regehr, G. R. Baker, R. Reznick, J. Bohnen, B. Orser, D. Doran, and E. Grober, "Communication failures in the operating room: an observational classification of recurrent types and effects," *BMJ Quality & Safety*, vol. 13, no. 5, pp. 330–334, 2004.

[17] M. J. Choi, J. J. Lim, A. Torralba, and A. S. Willsky, "Exploiting hierarchical context on a large database of object categories," in *Proc. CVPR*, 2010.

[18] S. H. Khan, X. He, M. Bennamoun, F. Sohel, and R. Togneri, "Separating objects and clutter in indoor scenes," in *Proc. CVPR*, 2015.

[19] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from RGB-D data," in *Proc. CVPR*, 2018.

[20] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE T-PAMI*, vol. 32, no. 8, pp. 1362–1376, 2010.

[21] J.-P. Pons, R. Keriven, and O. Faugeras, "Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score," *IJCV*, vol. 72, no. 2, pp. 179–193, 2007.

[22] R. A. Newcombe, O. H. S. Izadi, D. Molyneaux, D. Kim, A. J. Davison, P. Koh, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: real-time dense surface mapping and tracking," in *Proc. ISMAR*.

[23] S. M. S. Y. Furukawa, B. Curless and R. Szeliski, "Manhattan-world stereo," in *Proc. CVPR*, 2009.

[24] R. Cabral and Y. Furukawa, "Piecewise planar and compact floorplan reconstruction from images," in *Proc. CVPR*, 2014.

[25] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Inferring 3D object pose in RGB-D images," *arXiv preprint arXiv:1502.04652*, 2015.

[26] S. Tang, X. Wang, X. Lv, T. X. Han, J. Keller, Z. He, M. Skubic, and S. Lao, "Histogram of oriented normal vectors for object recognition with a depth sensor," in *Proc. ACCV*, 2012.

[27] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3D bounding box estimation," in *Proc. CVPR*, 2018.

[28] S. Song and J. Xiao, "Deep sliding shapes for amodal 3D object detection in RGB-D images," in *Proc. CVPR*, 2016.

[29] J. Xiao and Y. Furukawa, "Reconstructing the world's museums," *IJCV*, vol. 110, no. 3, pp. 243–258, 2014.

[30] S. Song, A. Zeng, A. X. Chang, M. Savva, S. Savarese, and T. Funkhouser, "Im2Pano3D: Extrapolating 360 structure and semantics beyond the field of view," in *Proc. CVPR*, 2018.

[31] L. Guo, G. Fan, and W. Sheng, "Robust object detection by cuboid matching with local plane optimization in indoor RGB-D images," in *Proc. VCIP*, 2017.

[32] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. ECCV*, 2014.

[33] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *ICRA*, 2011.

[34] R. K. Mahabadi, C. Häne, and M. Pollefeys, "Segment based 3D object shape priors," in *Proc. CVPR*, 2015.

[35] V. S. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *Proc. ICCV*, 2009.

[36] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proc. CVPR*, 2015.

[37] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE T-PAMI*, vol. 35, no. 8, pp. 1915–1929, 2013.

[38] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[39] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[40] M. Jian, C. Jung, and Y. Zheng, "Discriminative structure learning for semantic concept detection with graph embedding," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 413–426, 2013.

[41] Z. Deng and L. Jan Latecki, "Amodal detection of 3d objects: Inferring 3d bounding boxes from 2d ones in rgb-depth images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5762–5770, 2017.

[42] J. Zhang, Q. Wu, C. Shen, J. Zhang, and J. Lu, "Multilabel image classification with regional latent semantic dependencies," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2801–2813, 2018.

[43] J. Lahoud, B. Ghanem, M. Pollefeys, and M. R. Oswald, "3d instance segmentation via multi-task metric learning," *arXiv preprint arXiv:1906.08650*, 2019.

[44] C. Liu and Y. Furukawa, "Masc: Multi-scale affinity with sparse convolution for 3d instance segmentation," *arXiv preprint arXiv:1902.04478*, 2019.

[45] Y. Li, Y. Guo, J. Guo, Z. Ma, X. Kong, and Q. Liu, "Joint crf and locality-consistent dictionary learning for semantic segmentation," *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 875–886, 2018.

[46] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, 2016.

[47] A. H. Abdulnabi, B. Shuai, Z. Zuo, L.-P. Chau, and G. Wang, "Multimodal recurrent neural networks with information transfer layers for indoor scene labeling," *IEEE Transactions on Multimedia*, vol. 20, no. 7, pp. 1656–1671, 2017.

[48] X. Ding, B. Li, W. Xiong, W. Guo, W. Hu, and B. Wang, "Multi-instance multi-label learning combining hierarchical context and its application to image

annotation," *IEEE Transactions on Multimedia*, vol. 18, no. 8, pp. 1616–1627, 2016.

[49] M. Jian and C. Jung, "Semi-supervised bi-dictionary learning for image classification with smooth representation-based label propagation," *IEEE Transactions on Multimedia*, vol. 18, no. 3, pp. 458–473, 2016.

[50] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4490–4499, 2018.

[51] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "Panopticfusion: Online volumetric semantic mapping at the level of stuff and things," *arXiv preprint arXiv:1903.01177*, 2019.

[52] S. Ikehata, H. Yang, and Y. Furukawa, "Structured indoor modeling," in *Proc. ICCV*, 2015.

[53] Y. Z. M. B. P. Kohli, S. Izadi, and J. Xiao, "Deepcontext: Context-encoding neural pathways for 3D holistic scene understanding," *arXiv preprint arXiv:1603.04922*, 2016.

[54] M. J. Choi, A. Torralba, and A. S. Willsky, "A tree-based context model for object recognition," *IEEE T-PAMI*, vol. 34, no. 2, pp. 240–252, 2012.

[55] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *2011 IEEE intelligent vehicles symposium (IV)*, pp. 963–968, Ieee, 2011.

[56] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3523–3532, 2019.

[57] C. A. Vanegas, D. G. Aliaga, and B. Benes, "Building reconstruction using manhattan-world grammars," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 358–365, IEEE, 2010.

[58] M. Yazdanpour, G. Fan, and W. Sheng, "Online reconstruction of indoor scenes with local manhattan frame growing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[59] R. M. Lewis, V. Torczon, and M. W. Trosset, "Direct search methods: then and now," *JCAM*, vol. 124, no. 1, pp. 191–207, 2000.

[60] D. Lin, S. Fidler, and R. Urtasun, "Holistic scene understanding for 3D object detection with RGBD cameras," in *Proc. ICCV*, 2013.

[61] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.

[62] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," *arXiv preprint arXiv:1904.09664*, 2019.

[63] Z. Ren and E. B. Sudderth, "Three-dimensional object detection and layout prediction using clouds of oriented gradients," in *Proc. CVPR*, 2016.

[64] M. P. M. R. O. Jean Lahoud, Bernard Ghanem, "3d instance segmentation via multi-task metric learning," in *Proc. ICCV*, 2019.

[65] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, and N. Trigoni, "Learning object bounding boxes for 3d instance segmentation on point clouds," in *Proc. NIPS*, 2019.

[66] T. I. Y. K. Gaku Narita, Takashi Seno, "Panopticfusion: Online volumetric semantic mapping at the level of stuff and things," in *Proc. IROS*, 2019.

[67] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[68] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[69] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pp. 242–264, IGI global, 2010.

[70] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th international conference on Machine learning*, pp. 193–200, 2007.

[71] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang, "Heterogeneous transfer learning for image classification.," in *AAAI*, vol. 11, pp. 1304–1309, 2011.

[72] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[73] Y. Gao and K. M. Mosalam, "Deep transfer learning for image-based structural damage recognition," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 9, pp. 748–768, 2018.

[74] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *Advances in neural information processing systems*, pp. 2017–2025, 2015.

[75] H. Zhang, H. Zhang, C. Wang, and J. Xie, "Co-occurrent features in semantic segmentation," in *Proc. CVPR*, pp. 548–557, 2019.

[76] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. CVPR*, 2015.

[77] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3075–3084, 2019.

VITA

Lin Guo

Candidate for the Degree of

Doctor of Philosophy

Dissertation: Holistic Indoor Scene Understanding By Context Supported Instance Segmentation

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2020.

Completed the requirements for the Master of Science in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in July, 2015.

Completed the requirements for the Bachelor of Science in Electrical Engineering at Tianjin University, Tianjin, China in 2012.

Experience:

Visual Computing and Image Processing Lab (VCIPL), Oklahoma State University, Sep. 2014 – Dec. 2020

Research Assistant and Teaching Assistant, Oklahoma State University, Sep. 2014 – Dec. 2020

Professional Memberships:

IEEE Student Member 2017 - present