RGB-DEPTH IMAGE SEGMENTATION AND OBJECT RECOGNITION FOR INDOOR SCENES

A Dissertation
Submitted to
the Temple University Graduate Board

in Partial Fulfillment
of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

by

Zhuo Deng
2016

Examining Committee Members:

Longin Jan Latecki, Advisory Chair, Computer and Information Sciences
Haibin Ling, Computer and Information Sciences
Slobodan Vucetic, Computer and Information Sciences
Yimin Zhang, External Member, Electrical and Computer Engineering

ABSTRACT

With the advent of Microsoft Kinect, the landscape of various vision-related tasks has been changed. Firstly, using an active infrared structured light sensor, the Kinect can provide directly the depth information that is hard to infer from traditional RGB images. Secondly, RGB and depth information are generated synchronously and can be easily aligned, which makes their direct integration possible. In this thesis, I propose several algorithms or systems that focus on how to integrate depth information with traditional visual appearances for addressing different computer vision applications. Those applications cover both low level (image segmentation, class agnostic object proposals) and high level (object detection, semantic segmentation) computer vision tasks.

To firstly understand whether and how depth information is helpful for improving computer vision performances, I start research on the image segmentation field, which is a fundamental problem and has been studied extensively in natural color images. We propose an unsupervised segmentation algorithm that is carefully crafted to balance the contribution of color and depth features in RGB-D images. The segmentation problem is then formulated as solving the Maximum Weight Independence Set (MWIS) problem. Given superpixels obtained from different layers of a hierarchical segmentation, the saliency of each superpixel is estimated based on balanced combination of features originating from depth, gray level intensity, and texture information. We evaluate the segmentation quality based on five standard measures on the commonly used NYU-v2 RGB-Depth dataset. A surprising message indicated from experiments is that unsupervised image segmentation of RGB-D images yields comparable results to supervised segmentation.

In image segmentation, an image is partitioned into several groups of pixels (or super-pixels). We take one step further to investigate on the problem of assigning class labels to every pixel, i.e., semantic scene segmentation. We propose a novel image region labeling method which augments CRF formulation with hard mutual exclusion (mutex) constraints. This way our approach can make use of rich and accurate 3D geometric structure coming from Kinect in a principled manner. The final labeling result must satisfy all mutex constraints, which allows us to eliminate configurations that violate common sense physics laws like placing a floor above a night stand. Three classes of mutex constraints are proposed: global object co-occurrence constraint, relative height relationship constraint, and local support relationship constraint.

Segments obtained from image segmentation can be either too fine or too coarse. A full object region not only conveys global features but also arguably enriches contextual features as confusing background is separated. We propose a novel unsupervised framework for automatically generating bottom up class independent object candidates for detection and recognition in cluttered indoor environments. Utilizing raw depth map, we propose a novel plane segmentation algorithm for dividing an indoor scene into predominant planar regions and non-planar regions. Based on this partition, we are able to effectively predict object locations and their spatial extensions. Our approach automatically generates object proposals considering five different aspects: Non-planar Regions (NPR), Planar Regions (PR), Detected Planes (DP), Merged Detected Planes (MDP) and Hierarchical Clustering (HC) of 3D point clouds. Object region proposals include both bounding boxes and instance segments.

Although 2D computer vision tasks can roughly identify where objects are placed on image planes, their true locations and poses in the physical 3D world are difficult to determine due to multiple factors such as occlusions and the uncertainty arising from perspective projections. However, it is very natural for human beings to understand how far objects are from viewers, object poses and their full extents from still images. These kind of features are extremely desirable for many applications such as robotics navigation, grasp estimation, and Augmented Reality (AR) etc. In order to fill the gap, we addresses the problem of amodal perception of 3D object detection. The task is to not only find object localizations in the 3D world, but also estimate their physical sizes and poses, even if only parts of them are visible in the RGB-D image. Recent approaches have attempted to harness point cloud from depth channel to exploit 3D features directly in the 3D space and demonstrated the superiority over traditional 2D representation approaches. We revisit the amodal 3D detection problem by sticking to the 2D representation framework, and directly relate 2D visual appearance to 3D objects. We propose a novel 3D object detection system that simultaneously predicts objects' 3D locations, physical sizes, and orientations in indoor scenes.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Professor Longin Jan Latecki for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof.Haibin Ling, Prof.Slobodan Vucetic, and Prof.Yimin Zhang, for their insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

I thank my fellow labmates, Tianyang Ma, Meng Yi, Nan Li, Le Shu, Chen Shen, David Dobor, Ren-Hau Howard Liu, Cong Yao for the stimulating discussions, and for all the fun we have had in the last five years. Also I thank my roommates Xiao Jin, Yimin Jiao and friends Dawei Li, Ning Wang, Qiangnan Zhang, Min Xiao, Feipeng Zhao, Xin Li, Meng Ye, Liang Du, Yu Pang, Pengpeng Liang, Peiyi Li, Chao Han, Shanshan Zhang, Zhuangzhuang Gen, Wanhui Sheng, Mohammad Sharifian Gh and Fatemah Norouzi in the Temple University for the wonderful time we spent together.

Last but not the least, I would like to thank my family: my wife Mian Wang, my parents and parents in law for supporting me spiritually throughout writing this thesis and my life in general.

To My Wife - Mian Wang, My Parents and Parents in law

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

Figure

LIST OF TABLES

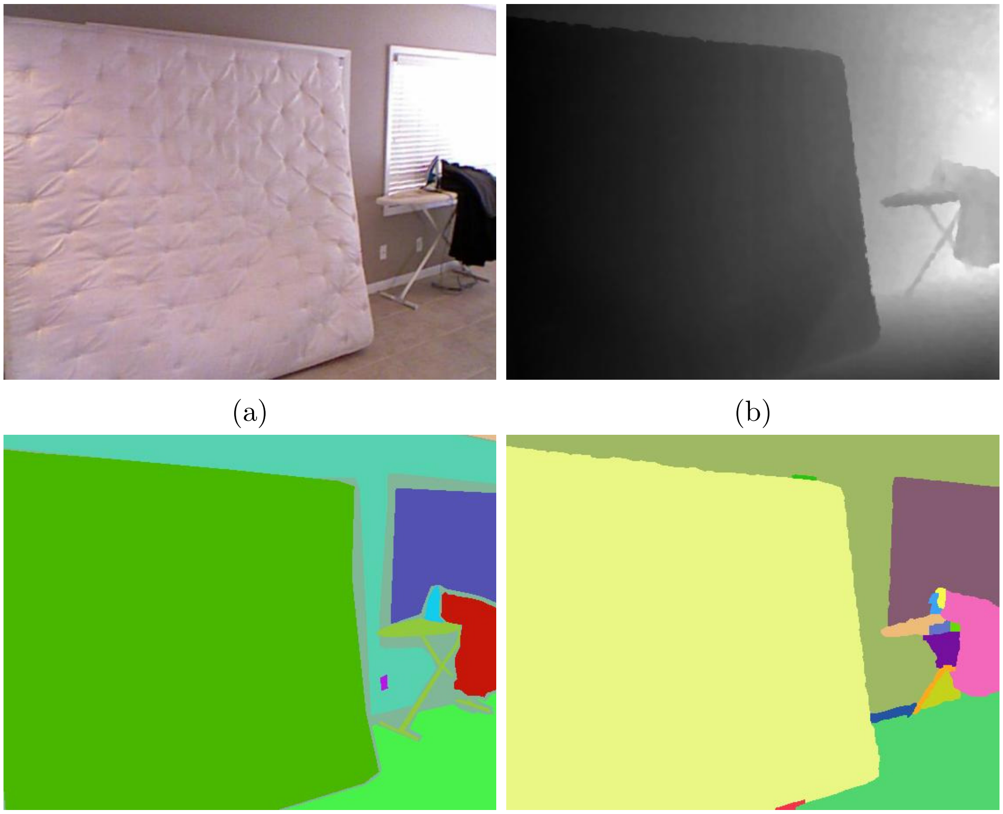
Chapter 1

RGB-Depth Unsupervised Image Segmentation

Unsupervised Image Segmentation (UIS) is one of the oldest and most widely researched topics in the area of computer vision, of which the goal is to partition an image into several groups of pixels that are visually meaningful using only the information provided by the single image.

In the past few decades, many great accomplishments have been made in this field from the early techniques [7, 43], which usually are based on the region splitting or merging framework to more recent works which tend to either integrate global constraints into grouping task, such as intra-region consistency and inter-region dissimilarity [18, 76, 1, 6], or formulate segmentation problem under clustering framework [10]. However, unsupervised image segmentation has remained an unsolved problem of computer vision, since RGB color information alone of a single image often does not provide sufficient information to successfully complete this task. There are many reasons for this, e.g., lack of distinctive features and instability of features due their sensitivity to illumination variation. Generally speaking, UIS is extremely difficult since incorrect segmentations (either too fine or too coarse) can be easily derived, even when employing algorithms that require the user to guess the number of segments.

Recently, with the advent of Microsoft Kinect, the landscape of various vision-related tasks has been changed. Firstly, using an active infrared structured light sensor, the Kinect can provide directly the depth information that is hard to infer from traditional RGB images. Secondly, RGB and depth information are generated synchronously and can be easily aligned, which makes their direct integration possible. A wide range of research works have demonstrated that RGB-D information is useful for improving the performance of vision tasks such as object recognition [56], scene labeling [79], body pose estimation [77], saliency detection [57] etc. The depth information itself is also very helpful for scene geometric structure estimation.



(a)

(b)

(c)

(d)

Figure 1.1: A typical indoor scene and our segmentation results. (a) Original RGB image obtained from Kinect camera. (b) Depth image, the missing values of which has been filled by the approach in [60]. (c) Ground truth segmentation. (d) Final segmentation result based on the proposed method.

The main goal of this paper is to explore the impact of RGB-D information on improving the unsupervised image segmentation. As we will demonstrate, the improvement is dramatic to the point that for many scenes the segmentation results are comparable to the results of supervised segmentation. Both supervised and unsupervised image segmentation that return a single scale complete image segmentation face the same problem of obtaining image segments correctly representing the scene objects of varying sizes. In particular, segments belonging to a single segmentation result may differ dramatically, some segments may fill nearly the whole image, representing objects like sofas in close view, and some may have area smaller that 1/100 of the image area. To solve this problem, we formulate the single scale segmentation as finding a maximum weight independent set (MWIS). This way we can automatically partition an RGB-D image into several salient regions with no need to specify either the number or sizes of regions in advance. A representative example is shown in Fig. 1.1.

The MWIS segmentation has been proposed for RGB images in [6]. It yields good segmentation results when foreground objects are very different from the background, since only then the region saliency measure is able to provide useful segment weights. Due to specific of RGB-D images, our saliency measure is very different and more informative. The main contribution of the proposed approach is a definition of region saliency measure that incorporates both RGB and depth information. As stated above such measure needs to properly balance the color and depth information, since for many objects only one of them is informative.

We test our method on the NYU depth dataset [79] and compare it to supervised hierarchical segmentation approaches in [79, 33]. [79] starts from an over-segmentation, and adapts the algorithm in [42] to iteratively merge regions based on boundary strength. This approach is supervised, since the boundary strength needs to be learned from labeled instances. Similarly, [33] trains oriented contour detectors based on features extracted from watershed over-segmentation contours. Finally, initial over-segmentation regions are merged based on the average strength of oriented contour detectors. Although our method is unsupervised, it obtains comparable results to [79, 33]. Moreover, we also compare our approach to an unsupervised segmentation method in [84]. It extends the work of [18] by creating an extra edge on the original graph, of which the weight is measured based on the angle difference of surface normals obtained from depth information. In addition, we also use gpb-owt-ucm as a baseline where depth information is not used. We evaluate the segmentation quality based on five standard measures: Probabilistic Rand Index (PRI) [89], Variation of Information (VI) [66], Global Consistency Error (GCE) [65], Boundary Displacement Error (BDE) [22] and Jaccard Index (JI)[1]. Our approach significantly outperforms [84] in all five measures, which clearly demonstrates the superiority of the proposed combination of color and depth information.

1.1 Related Works

Image segmentation is a fundamental problem and has been studied extensively. Classic image segmentation approaches include normalized cuts [76], minimum spanning tree [18], meanshift [10], and gPb-OWT-UCM[1]. However, these approaches can only obtain segmentation results comparable to humans if their parameters are known in advance or in other words manually tuned. For example, the normalized cuts requires assigning a specific number of regions at the beginning. Therefore, these algorithms are usually run with different parameter settings, which yields multi-scale image segmentation results. While multi-scale results are very useful for many supervised methods for object detection, scene labeling or image segmentation, it is hard to utilize them to obtain a single segmentation result of an RGB image in unsupervised setting.

One common drawback of these unsupervised segmentation techniques is that they have no prior knowledge about the geometric structure of the scene, which leads to the segmentation to be either too coarse if two spatially separated regions have similar appearance or too fine when one planar region contains subregions with different textures. Although recent approaches that try to infer the 3D structure of the scene given only a single RGB image, e.g., [41, 38, 39, 58, 30], they are limited to very simple structures.

The emergence of the RGB-D technology provides a great opportunity to take advantages of merits from both RGB and depth information. Some of the recent works on unsupervised RGB-D segmentation integrate the image segmentation with plane fitting [29, 16]. In [29], the RGB-D segmentation is formulated as iterative refinement of the pixel-to-plane assignment and optimized as discrete labeling in a Markov Random Field (MRF), with plane merging controlled by a threshold. [16] formulates the plane fitting as a linear least-squares problem and infers the segmentation of the scene in a Bayesian framework. The other unsupervised segmentation works are trying to adapt the classic segmentation algorithms into the RGB-D field. [86] first detects edges on RGB images and computes triangular tessellation of images based on edge information by the Delaunay Triangulation algorithm. Then a variant of N-cut is applied to the graph constructed from the triangular regions. Finally the segments from N-cuts are used to suggest groupings of depth samples from depth image. [87] extends the work in [86] to segment the Manhattan structure of an indoor scene from a single RGB-D frame into floor plane and walls. In contrast to these approaches, our method is not limited to planar structures in the scene. Similar to our work, in [45], image segmentation is formulated as finding high-scoring maximal weighted cliques in a graph connecting non-overlapping putative figure-ground segment hypothesis. In [59], the pylon model is proposed to find a globally optimal subset of segment pool and their labels through graph-cuts and max-margin learning. But both [45] and [59] are supervised whereas ours is an unsupervised method. Except for unsupervised segmentation, supervised segmentation also benefits from the RGB-D technology. One of the most recent works is [79], where regions with minimum boundary strength are iteratively merged in a hierarchical framework. The boundary is predicted by a trained boosted decision tree classifier based on labeled instances. The other one proposed in [33] utilize depth information to train several oriented contour detectors. Hierarchical segmentation is constructed by merging regions of initial over-segmentation based on the average strength of those oriented contour detectors. Unlike the above works, the proposed approach is completely unsupervised, since it does not require any parameter learning from labeled instances, nor we make any assumptions about the number of regions to be segmented.

1.2 General Framework

1.2.1 Hierarchical image segmentation

To partition one image into superpixels, there are several excellent algorithms such as the gPb-OWT-UCM method of [1], the minimum spanning tree segmentation [18], the multi-scale normalized cuts [12], mean shift segmentation [10], and watershed based segmentation [67]. In this paper, we adapt the method introduced in [1] to integrate both RGB and depth information for hierarchical segmentation. In [1], firstly an over-segmentation is derived based on the watershed transformation of the gradient map, which is a linear combination of brightness, color, texture gradients and spectral signal. Following the multiple cues combination framework, we integrate depth and normal gradients directly into the final gradient map. Suppose we denote an image as $I(x, y)$, the gradient map $G(x, y)$ is represented as

$$G(x, y) = w_b G_b + w_c G_c + w_d G_d + w_n G_n + w_s G_s, \tag{1.1}$$

where G_b and G_c are brightness and color gradient signals respectively, which are computed in the CIE-LAB color space. G_d is the gradient signal estimated based on depth image. G_n represents the normal signal where the difference of two normal vectors $\mathbf{n}_i$ and $\mathbf{n}_j$ is measured as

$$Dist(\mathbf{n}_i, \mathbf{n}_j) = sin(acos(\frac{\mathbf{n}_i \bullet \mathbf{n}_j}{|\mathbf{n}_i||\mathbf{n}_j|}), \tag{1.2}$$

and G_s is the spectral signal. All the gradient signals except for the spectral signal are estimated by convolving a 3×3 sobel kernel with signals themselves. Then an over-segmentation is obtained by applying the watershed transformation to $G(x, y)$. In order to present the hierarchical segmentation, Ultrametric Contour Map (UCM) is used to capture the average strength of shared boundary between two adjacent regions based on $G(x, y)$. For an input RGB-D image, we obtain 7 scales of hierarchical image segmentation by adjusting the strength threshold θ_g on the UCM. We denote with V the set of all superpixels from all scales and from both RGB and D images.

1.2.2 Saliency measure of superpixels

The goal of this section is to compute the saliency measure for each superpixel in V. For RGB-D segmentation, a critical issue is how to integrate depth information with RGB information in order to obtain a weight of each superpixel. Previous works such as [71] and [29] assign a fixed importance weight to RGB and depth information respectively based on parameter training or empirical setting. However, it is not the case that depth information is more important than RGB information nor vice versa. In reality, when we are trying to identify a salient object from its background, the criteria used always change. For example, based on depth it is easy to separate the surface of a desk from the floor. Whereas, to distinguish a bedsheet from a bed frame, color or texture properties are more helpful. Based on this intuition, we propose a novel weighting scheme to estimate the saliency of superpixels in RGB-D images.

We estimate the saliency by combining three kinds of information: depth, gray level intensity, and textures. Suppose we denote a superpixel as $S_i \in V$ and given depth image $I_d(x, y)$, and RGB image $I_c(x, y)$. We extract gray scale image $I_g(x, y)$ from $I_c(x, y)$. The corresponding saliency measures $C_d(S_i)$, $C_g(S_i)$, $C_t(S_i)$ are defined below. The higher their values, the more uniform is superpixel S_i. We define the saliency of superpixel S_i as their weighted average

$$w(S_i) = W_{area}(w_1 C_d(S_i) + w_2 C_g(S_i) + w_3 C_t(S_i)), \tag{1.3}$$

where $w_1, w_2, w_3 \geq 0$, $w_1 + w_2 + w_3 = 1$,

$$W_{area} = (1 - \exp(-\eta \frac{|S_i|}{|I(x, y)|}))$$

is used to slightly favor larger regions. The weights w_1, w_2, w_3 are dynamically assigned so that the value of most informative of the three saliency measures $C_d(S_i)$, $C_g(S_i)$, $C_t(S_i)$ has the higher weight. We have three constant values $\alpha > \beta > \gamma > 0$ for the weights and assign the largest value to the largest feature, e.g., if $C_d(S_i) > C_g(S_i) > C_t(S_i)$, then $w_1 = \alpha, w_2 = \beta, w_3 = \gamma$.

Unlike [57] where the relationship between saliency and depth is trained by fitting a GMM, we directly define the confidence from depth information $C_d(S_i)$

as

$$C_d(S_i) = \exp(\frac{-std(\{G_d(p)|p \in S_i\})}{|\underset{p\in S_i}{avg}(\{I_d(p)\}) - \underset{p\in S^i_{ext}}{avg}(\{I_d(p)\})|}) \tag{1.4}$$

where $p = (x, y)$ represents a pixel at position (x, y), S^i_{ext} denotes the neighboring area of S_i, and $G_d(x, y)$ represents the gradient map of $I_d(x, y)$. This term encourages the planar region that has high contrast to its surrounding area on the depth value.

The gray scale confidence is defined as

$$C_g(S_i) = \exp(\frac{-\underset{p\in S_i}{std}(\{I_g(p)\})}{\underset{p\in S^i_{ext}}{std}(\{I_g(p)\})}). \tag{1.5}$$

The region where pixels have similar intensity value within it and dissimilarity is high with respect to its neighbor area should be assigned a heavier weight.

In order to estimate the weight from the texture perspective, we firstly apply the Maximum Response (MR8) filter bank [90] to the gray scale image $I_g(x, y)$. MR8 filter bank consists of 38 filters (6 orientations at 3 scales for 2 oriented filters and 2 isotropic filters) and the number of filter responses is reduced to eight. Each pixel of $I_g(x, y)$ is attached with a filter response vector $\mathbf{f}_r$. Then K-means clustering are used to extract k "vector words". Each vector $\mathbf{f}_r$ is assigned an integer label of the "vector word" which is closest. In order to measure the texture saliency, we use the J-measure proposed in [14] that is based on spatial distributions of pixels of similar properties. Suppose there are n_c different labels in S_i, C_i denotes all pixels in S_i with the same quantized label, and N_i is the number of pixels in C_i. The center of C_i is denoted as $m_i = \frac{1}{N_i}\sum_{p\in C_i} p$. We define

$$S_W = \sum_{i=1}^{n_c} \sum_{p\in C_i} ||p - m_i||^2 \tag{1.6}$$

and observe that S_W is small if there are compact clusters of labels in S_i while it is large if pixels with different labels are uniformly distributed in S_i. We also define the spread of all pixels in S_i as

$$S_T = \sum_{p \in S_i} ||p - m||^2 \tag{1.7}$$

where m is the central point of S_i. The texture salience is then defined as

$$C_t(S_i) = \exp(\frac{S_W - S_T}{S_W}) \tag{1.8}$$

If all the pixel labels are distributed uniformly over the entire superpixel area, the value of $C_t(S_i)$ is large. In contrast, it is small if there are compact clusters of labels in S_i.

1.2.3 Final Segmentation as MWIS

We first construct a graph composed of superpixels $S_i \in V$ as its nodes, where $|V| = n$ We assign to each node $S_i \in V$ a weight $w_i = w(S_i)$ defined in formula (1.3). We observe that all weights are nonnegative and denote with $\mathbf{w} = [w_1, w_2, ..., w_n]^\top$ the weight vector.

The adjacency matrix M is defined as follows. An edge exists between two superpixels S_i and S_j if they overlap, i.e., $M_{ij} = 0$ if $S_i \cap S_j = \emptyset$ and $M_{ij} = 1$ otherwise. We obtain an undirected graph $G = (V, M, \mathbf{w})$.

In graph theory, an *independent* set is a set of vertices in a graph where no two vertices are adjacent. The *maximal independent set* is an independent set which has the largest number of vertices. In the case we have a weight attached to each vertex, the *maximum weight independent set (MWIS)* is an independent set with the largest sum of the node weights.

An indicator vector, $\mathbf{x} = [x_1, x_2, ..., x_n]^\top \in \{0, 1\}^n$, is used to denote any subset B of the graph nodes, where $x_i = 1$ means node $S_i \in B$ and $x_i = 0$ means node $S_i \notin B$. When B is an independent set and $\mathbf{x}$ its indicator vector, we have $\forall(i,j), x_i \cdot x_j = 0$ if $M_{ij} = 1$. Hence it holds that $\mathbf{x}^\top M\mathbf{x} = 0$. Therefore, $\mathbf{x}^*$ representing the MWIS can be obtained as the solution of the following quadratically constrained integer linear program

$$\begin{aligned} &\mathbf{x}^* = \underset{\mathbf{x}}{argmax} \;\; \mathbf{w}^\top\mathbf{x} \\ &s.t. \; \forall i \in V : x_i \in \{0,1\}, and \;\; \mathbf{x}^\top M\mathbf{x} = 0 \end{aligned} \tag{1.9}$$

We solve the program (1.9) with the algorithm introduced in [6]. The solution vector $\mathbf{x}^*$ selects superpixels that compose our final single scale segmentation of a given image.

1.3 Experiments

This section presents both qualitative and quantitative evaluation of our unsupervised segmentation algorithm on 1449 pairs of aligned RGB and depth images from the NYU Depth Dataset V2 [79]. Detailed ground truth segmentation is provided for each image. This data set is very challenging for segmentation, even with RGB-D information, because of poor illumination, often rendering RGB information useless, cluttered non-planar stuff (eg. bedsheets, sofa, clothes etc), which strongly limits the depth cues, large variation of scene types, and non-perfect depth measurement. In particular, depth images contain "black holes" due to missing data, and random error of depth measurements increase quadratically with the increasing distance from the sensor [50]. Also the average density of depth measurements decreases when the distance to the objects increases, since the resolution of Kinect is fixed at $480 * 640$.

In order to evaluate our algorithm quantitatively, five standard evaluation measures are employed. The first one is Probabilistic Rand Index (PRI), which estimates the ratio between pairs of pixels, whose labelings are consistent in both ground truth and estimated segmentation, and the total number of pixel pairs. Variation of Information (VI) measures the distance between two segmentations by the average conditional entropy of one segmentation given the other. Global Consistency Error (GCE) measures the extent to which one segmentation can be viewed as a refinement of the other. The Boundary Displacement Error (BDE) measures the average displacement error of boundary pixels between two segmented images. Particularly, it defines the error of one boundary pixel as the distance between the pixel and the closest pixel in the other boundary image. The Jaccard Index (JI) measures similarity between two segmentations, and is defined as the size of the intersection divided by the size of the union of the two segmentations.

We first compare our method to the two baseline UIS methods: in [1], depth information is not used and in [84], normal vector information is applied. For [1], we select the best layer from the hierarchical segmentation based on the five evaluations. As can be seen in Table 1.1, our method significantly outperforms both of the baseline methods on all five evaluation measures. Surprisingly, the result of [1] is slightly better than [84]. We also compare our approach to two recent RGB-D supervised segmentation methods proposed in [79, 33]. Therefore, following the same dataset split setting, training set contains 795 images, and performance is evaluated on 654 test images. Since the algorithm in [79] outputs a hierarchical segmentation composed of five segmentation levels, we choose the best result based on the five standard evaluation measures out of the five levels for each image. [33] similarly outputs a hierarchical segmentation of 99 segmentation levels. We use the best layer as evaluated in their paper (threshold = 0.54). Although our method is unsupervised, for fair comparison, we also evaluate it on the 654 test images. As can be seen in Table 1.1, the performance of our method

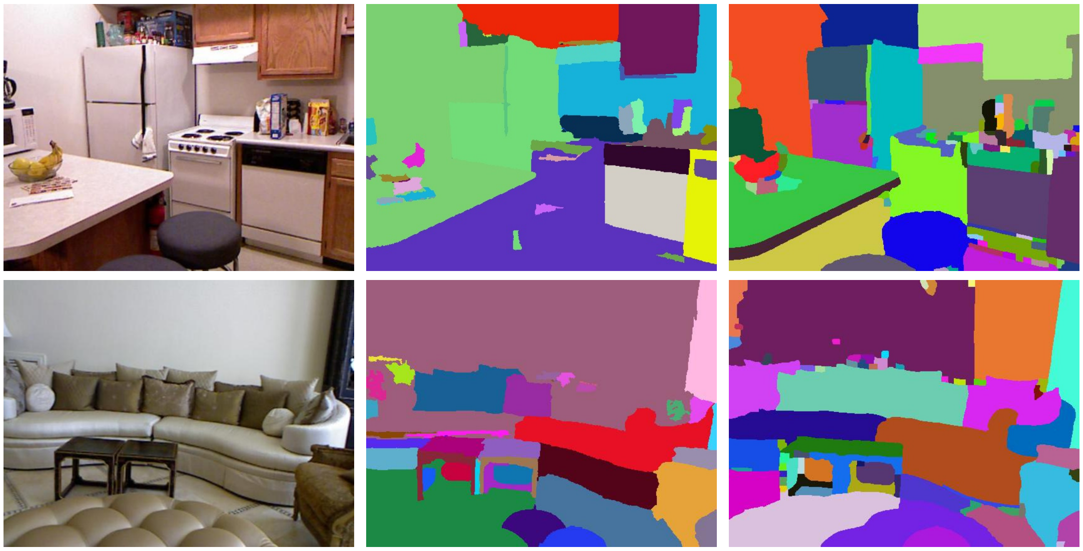


Figure 1.2: Two examples to illustrate the benefits of using depth information. The first column contains two original RGB images from Kinect. The second column is the segmentations only based on RGB information. The third column contains the corresponding segmentations based on both RGB and depth information.

is very close to theirs. This is very surprising for at least three reasons: 1) Our method is unsupervised, while the method in [79, 33] are supervised. 2) Our method is much simpler than the methods in [79, 33]. 3) Our segmentation result sometimes shows more details than the ground truth, since it is not restricted to known object classes, which incorrectly lowers our accuracy.

In order to visually compare supervised segmentation results [79, 33] with our unsupervised segmentation results, we list 8 different samples in the Fig. 1.3. in varieties of scene categories such as bookstore, living rooms, offices, classrooms and so forth. As can be seen the segmentation of our result is very competitive. Our approach is robust to the variation of illumination, even when scenes are dark (eg. the scene in the bathroom) or when scenes are extremely bright, e.g., the blinds of the living room in Fig. 1.1 and the surface of the blackboard in the conference room, or when shades are projected on objects, e.g., the shades on

Method	PRI	GCE	VI	BDE	JI
RGB [1]	0.889	0.178	2.253	9.236	0.527
RGBD [84]	0.875	0.298	2.165	11.381	0.488
RGBD [79]	**0.917**	0.122	1.706	**7.509**	0.605
RGBD [33]	0.916	0.162	**1.501**	7.808	**0.622**
Ours RGBD	0.914	**0.120**	1.891	8.488	0.583

Table 1.1: Segmentation accuracy evaluated on 654 test RGB-D images in the NYU Depth Dataset V2 [79], since methods in [79] and [33] are supervised. The values are: PRI (larger is better), VI (smaller is better), GCE (smaller is better), BDE (smaller is better) and JI (larger is better).

the floor and wall of the bedroom scene. Our approach also works well in very cluttered indoor scenes, like the scenes in the bookstore and the lady's office.

The results in Fig. 1.2 also demonstrate that depth information is really helpful in our framework for distinguishing objects with similar colors but different locations from each other. As can be seen in the kitchen scene, the surface of the table, the wall, and the refrigerator have similar white color, and in the living room scene, the sofa and the blanket on the floor also have similar color. So when only RGB information is used, different objects are inclined to be segmented as one superpixel. However, when the depth information is added, all of them become correctly separated.

The average run time per image segmentation is listed in Table 1.2. It was evaluated on a PC computer with AMD Eight-core CPU @ 3.1HZ and 16GB RAM. Except for [84] which runs in C++, our method is much faster than GPb-OWT-UCM and other two supervised methods.

Parameter setting: The input to our segmentation are superpixels obtained from hierarchical segmentations. As is mentioned in Section 1.2.1, we obtain segmentations at different levels by changing the value of the strength threshold θ_g

[79] in Matlab	our method in Matlab	[84] in C++	[1] in Matlab	[33] in Matlab
122.1	68.8	7.39	301.1	> 300

Table 1.2: The average run time in seconds to segment a single image.

which falls between 0 and 1. When θ_g increases, the number of regions segmented is reduced. Experimentally, we find that if the segmentation in each layer is too fine, it may produce many areas that consists of only several pixels. They are not only meaningless but also tend to increase the burden of computation. On the other hand, if the segmentation in each layer is too coarse, it also can not provide good candidate regions. Therefore, we set the θ_g to $[0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6]$. For the weights of different gradient signals, we simply set them as $w_b = 1.0$, $w_c = 0.5$, $w_n = 3.0$, $w_d = 2.0$ and $w_s = 3.0$ since depth information and global spectral signal are much more reliable than brightness and color. In addition, we set constants α, β, and γ to 0.5, 0.3, 0.2 respectively. The constant η is set to 10 in our experiment.

1.4 Conclusion

In this paper, we propose an unsupervised segmentation method for RGB-D image segmentation. It integrates both color and depth information effectively and partitions one RGB-D image into several most salient regions without the need to know the number or the size of segments in advance. Our experiments on the NYC depth dataset show that the segmentation accuracy of our method is very competitive with respect to both unsupervised and supervised methods. Also the fact that our method is very efficient due to its simplicity, makes it very suitable for many applications from object to action recognition.

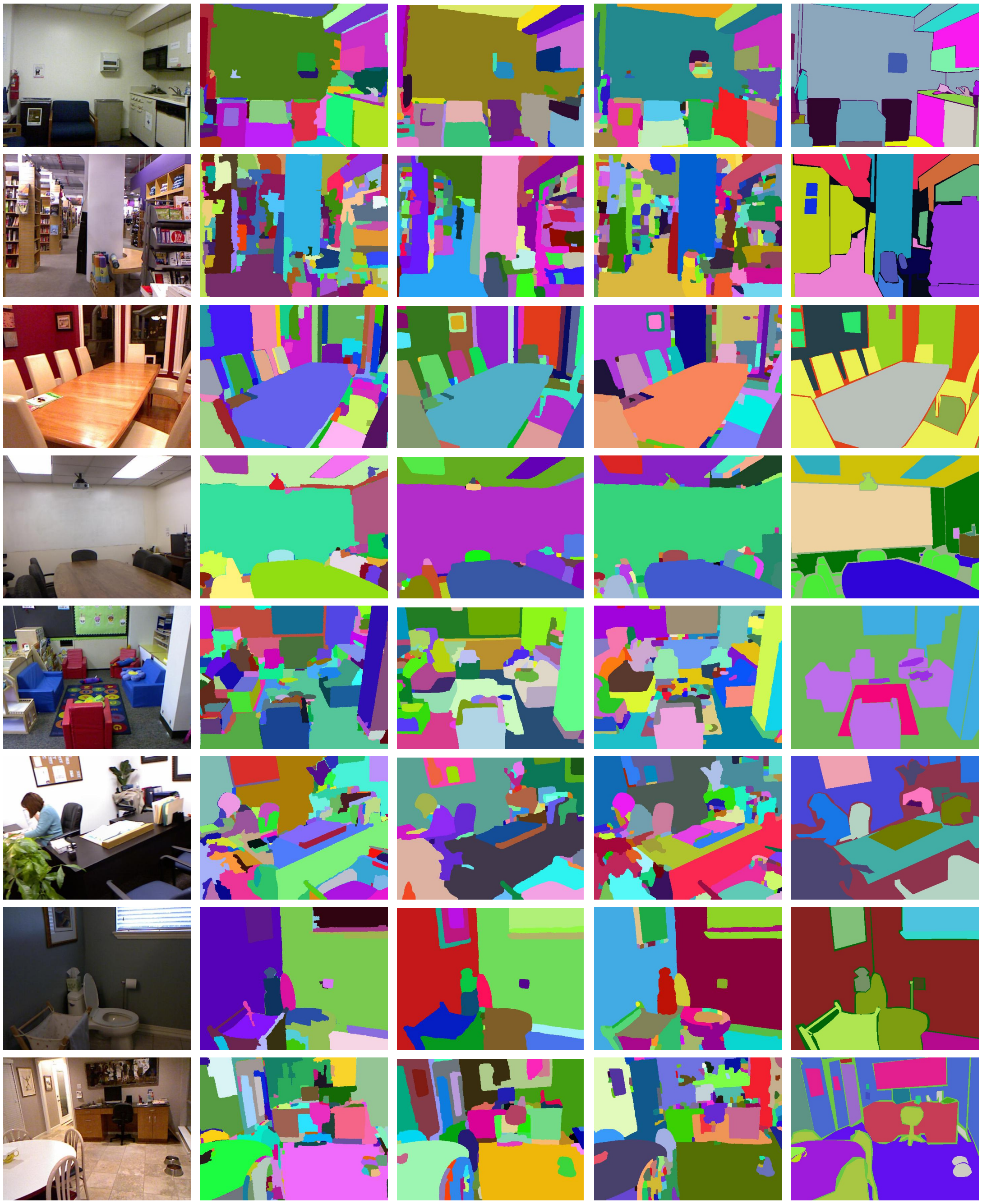


Figure 1.3: Examples of unsupervised indoor scene segmentation obtained by our method and supervised methods in [79, 33]. Column 1 shows the original RGB images. Column 2 shows results in [79]. Column 3 shows results in [33]. Column 4 shows our segmentations and last column shows the ground truth.

Chapter 2

Semantic Segmentation of RGBD Images with Mutex Constraints

2.1 Introduction

This paper addresses the fundamental problem of semantic scene segmentation of indoor scenes. Assigning class labels to every pixel in real-world images is challenging, as objects may be heavily occluded, appear in a wide range of configurations, and viewed from different camera viewpoints and distances. In addition, indoor scenes typically consist of a relatively large number of alike objects that are often cluttered and in disorder, reflecting various lifestyles. Our goal is to partition the image by identifying subimage ownership among occurrences of distinct object classes.

The recent advent of Microsoft Kinect alleviated some of these challenges, and thus enabled an exciting new direction of approaches to semantic scene segmentation [71, 79, 33, 78, 11, 49, 34, 54]. Equipped with an active infrared structured light sensor, Kinect is able to provide the depth information of objects in the scene which is aligned synchronously with their color images. Since indoor scenes are typically characterized by large planar surfaces (e.g., floor, walls, table tops), and objects can often be interpreted in relation to those surfaces, semantic scene segmentation can be largely facilitated by properly integrating visual cues with detailed and accurate geometric structure of the scene surfaces provided by Kinect.

Recent work has demonstrated that the depth information can be readily used to leverage rich geometric structure of indoor scenes toward their robust semantic segmentation. The SLAM technology was used to merge multiple RGBD images into a single point cloud and densely label it with Markov Random Field (MRF) [54]. Scenes were labeled by incorporating SIFT features and 3D location priors into a Conditional Random Field (CRF) [78]. A CRF with higher order cliques was used to encourage all regions in them to take the dominant label [49]. [71]

extended the Kernel Descriptors (KDES) [3] by introducing depth gradient and spin normal descriptors, and labeled scenes by combining MRF with segmentation tree. In [33], geometric features were integrated with traditional visual features through support vector machines, or with high level features from object detection [34]. Instead of designing hand crafted features, a multiscale convolutional network was used to learn features directly from RGBD images [11].

Although designing distinct features from RGBD images has achieved much progress for indoor semantic segmentation, how to jointly model local and long range object spatial configurations by taking advantage of available geometric structure of indoor scenes is not fully explored. We find that there is still room for improvement.

In this paper, we propose a holistic framework for reasoning about object classes and their co-occurrences, and spatial layouts based on geometric structure of indoor scenes as well as on common sense knowledge. We model the scene by a CRF grounded to regions of the low-level generalized gPb-UCM segmenter [33]. Geometric and visual information of objects are integrated into unary potentials. The pairwise potentials encode local object configurations based on several typical geometric patterns. In this way, we pose semantic scene segmentation as the problem of assigning class labels to image regions in the CRF inference. As common, we cast CRF inference as a quadratic programming (QP) problem.

As our key contribution, we incorporate in our QP *qualitative* common-sense constraints from domain knowledge in a principled manner. We focus on mutual exclusion (mutex) constraints that specify *negation* (inconsistency) rules about object configurations in the real physical world. For example, a chair should *not* be on top of a TV, and a floor should *not* occur above a dishwasher. In scene labeling, mutex constraints are binary relations specifying inconsistent class label assignments to pairs of image regions, and can be expressed without any higher-order potentials. Also, model expressiveness is significantly increased as they can enforce *long-range* consistency constraints on the solution. With mutex constraints, our approach can make use of rich and accurate geometric structure coming from Kinect in a principled manner.

Prior works have already demonstrated the importance of domain constraints, in general, as they help resolve competing hypotheses when visual cues are not sufficient for scene interpretation. Constraints are typically incorporated in CRFs as features of the pairwise potentials [28, 23, 79]. More sophisticated methods use higher-order constraints, beyond pairwise [53, 55, 49]. Instead of working our way through higher-order constraints, we focus on exclusion common-sense rules, i.e., hard rules that exclude nonsense configurations.

In this paper, we show that mutex constraints can be compactly expressed in a quadratic equality form, and rigorously enforced in a principled manner. As smoothness and constraints are typically combined in the pairwise potential, traditional formulations of CRF inference may not guarantee that hard constraints are all satisfied. This could yield non-sensical results. We address this problem by expressing the mutex constraints as quadratic constraints of our QP. The most closely related works are [74] and [64], both of which utilize mutex relations to constrain the CRF inference. However, both [74] and [64] work with 2D images only. The goal of [74] is foreground object segmentation in videos, while [64] is focused on scene labeling. In contrast, the focus of our work is on 3D mutex relations representing common sense knowledge. Since understanding RGB-D indoor scenes is an arguably more complex task [71, 70], in addition, we utilize 3D geometric patterns and spatial object correlation for edge potential estimation, instead of the standard Potts model in [74]. Moreover, we are using a sparsely connected CRF model.

In this paper, we empirically demonstrate that enforcing *qualitative* mutex constraints can significantly improve quantitative measures of performance. The effectiveness of our approach is evaluated on the indoor scene NYU dataset V2 [79] and a recent SUN3D dataset [93]. Our labeling accuracy significantly outperforms the state of the art [33, 34].

In the rest of this paper: Sec. 2.2 formulates our CRF model and CRF inference as QP for semantic segmentation; Sec. 2.3 specifies unary and pairwise potentials that are used to compute the affinity matrix for our QP; Sec. 2.4 describes how to estimate mutex constraints from training data; and Sec. 3.3 presents experimental results and related discussion.

2.2 CRF for Semantic Segmentation

This section formulates our CRF model of a scene grounded on low-level segments (also called superpixels), and casts semantic segmentation as the MAP assignment of class labels to superpixels. We begin by specifying the quadratic objective of the MAP assignment problem, and then extend that formulation to include mutex constraints, resulting in our integer QP with quadratic constraints.

2.2.1 CRF and the MAP Assignment as QP

As in [79, 33, 34], we partition an image, $I(x, y)$, into a set of segments $\mathbb{S} = \{s_i : i = 1, \ldots, N\}$, $|\mathbb{S}| = N$, using variants of the gPb-UCM hierarchical segmentation algorithm [1]. Each segment, $s_i \in \mathbb{S}$, can take one object class label, l_i, from the set of labels $l_i \in \mathbb{L}$, $|\mathbb{L}| = L$. Each label assignment to a superpixel, (s_i, l_i), can be represented as a node of the association graph $\mathbb{G} = (\mathbb{V}, \mathbb{E}, A)$, where $\mathbb{V} = \mathbb{S} \times \mathbb{L}$ is the set of nodes, $|\mathbb{V}| = N \cdot L$, and $\mathbb{E} \subset \mathbb{V} \times \mathbb{V}$ is the set of graph edges. We define $((s_i, l_i), (s_j, l_j)) \in \mathbb{E}$ if s_i and s_j are *spatially adjacent*, which means that their shared boundary in $I(x, y)$ contains at least one pixel and the minimal 3D distance between point clouds projecting to s_i and s_j is very close.

A is the adjacency matrix (or the affinity matrix) of $\mathbb{G}$, with size $(N \cdot L) \times (N \cdot L)$.

We define a CRF over $\mathbb{G}$. To this end, we associate a latent binary random variable $X_{s_i,l_i} \in \{0, 1\}$ with every node $(s_i, l_i) \in \mathbb{V}$. When X_{s_i,l_i} is instantiated to value $x_{s_i,l_i} = 1$ then the CRF assigns class label $l_i \in \mathbb{L}$ to superpixel $s_i \in \mathbb{S}$. The column vector of all instantiations of the assignment random variables is denoted as $\mathbf{x} = [\ldots, x_{s_i,l_i}, \ldots]^\top \in \{0, 1\}^{N \cdot L}$.

We use the affinity matrix A to specify the unary and pairwise potentials of the conditional log-likelihood of the CRF. In particular, the diagonal elements $A((s_i, l_i), (s_i, l_i))$ encode the unary potentials corresponding to log-likelihoods of label assignments $x_{s_i,l_i} = 1$. The off-diagonal elements $A((s_i, l_i), (s_j, l_j))$ encode the pairwise potentials corresponding to joint log-likelihoods of label assignments $x_{s_i,l_i} = 1$ and $x_{s_j,l_j} = 1$.

From above, the conditional log-likelihood of the CRF is specified as

$$\begin{aligned} \log P(\mathbf{x}|\mathbb{G}) = & \sum_{(s_i,l_i)\in V} A((s_i, l_i), (s_i, l_i)) x_{s_i,l_i} \\ + & \sum_{((s_i,l_i),(s_j,l_j))\in E} A((s_i, l_i), (s_j, l_j)) x_{s_i,l_i} x_{s_j,l_j} - Z(\mathbb{G}), \end{aligned} \tag{2.1}$$

where $Z(\mathbb{G})$ is the partition function.

From (2.1), it follows that the semantic scene segmentation problem can be formulated as finding the MAP assignment $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\Omega} P(\mathbf{x}|\mathbb{G})$, where Ω is the space of allowed solutions. Note that the MAP assignment is independent of $Z(\mathbb{G})$. Thus, we can compactly express the MAP assignment problem as the following integer QP with linear constraints:

$$\begin{aligned} \text{QP-L}: \quad & \text{maximize} \quad \mathbf{x}^\top A \mathbf{x} \\ \text{s.t.} \quad & \text{for all} \;\; s_i \in \mathbb{S}, \sum_{l_i\in\mathbb{L}} x_{s_i,l_i} = 1, \;\; \mathbf{x} \in \{0, 1\}^{N \cdot L}. \end{aligned} \tag{2.2}$$

The linear constraints in the QP-L, given by (2.2), ensure that every superpixel in the image gets assigned a unique class label. In the following, we extend QP-L such that the resulting QP encodes mutex constraints.

2.2.2 QP with Mutex Constraints

This section formulates mutex constraints in a quadratic equality form, combines them with the linear constraints of QP-L, and thus expresses the MAP assignment problem as an integer QP with quadratic equality constraints.

Mutex constraints are aimed at prohibiting certain non-sensical label assignments to superpixels in the image. We eliminate this hypothesis by enforcing $x_{s_i,l_i}{\cdot}x_{s_j,l_j}{=}0$. That is, only one of the two label assignments is allowed. If one is accepted as a solution then it automatically prevents the other one. Using the notation introduced in Sec. 2.2.1, it follows that all mutex constraints can be compactly represented as

$$\text{Quadratic mutex constraints (QMC)}: \quad \mathbf{x}^\top M\mathbf{x} = 0, \tag{2.3}$$

where M is a $(N \cdot L) \times (N \cdot L)$ binary mutex matrix. Note that when matrix elements are set to one, $M((s_i, l_i), (s_j, l_j)) = 1$, then the corresponding assignments are prohibited and hence $x_{s_i,l_i} = 0$ and/or $x_{s_j,l_j} = 0$ in order to enforce $x_{s_i,l_i}{\cdot}1{\cdot}x_{s_j,l_j}{=}0$. Conversely, when $M((s_i, l_i), (s_j, l_j)) = 0$ then superpixels s_i and s_j may be assigned any class labels, because $x_{s_i,l_i} \cdot 0 \cdot x_{s_j,l_j} = 0$. If the sum of each row of M is at least one, then M represents global mutex constraints. This means that at least one constraint applies to each variable.

Further, it is convenient to merge the set of linear constraints of QP-L — namely that for all $s_i \in \mathbb{S}$, $\sum_{l_i \in \mathbb{L}} x_{s_i,l_i} = 1$ — with the quadratic mutex constraints (QMC) in (2.3). For every superpixel s_i, we set all matrix elements

$M((s_i, l_i), (s_i, l_i')) = 1$, if $l_i \neq l_i'$. This prohibits illegal assignments of two (or more) distinct labels to a single superpixel.

From (2.2) and (2.3), we finally derive the MAP assignment problem as the integer QP with quadratic constraints:

$$\begin{aligned} \text{QP-Q}: \quad & \text{maximize} \quad \mathbf{x}^T A \mathbf{x} \\ & \text{subject to} \quad \mathbf{x}^{\mathbf{T}} M \mathbf{x} = 0 \quad , \quad \mathbf{x} \in \{0,1\}^{N \cdot L}. \end{aligned} \tag{2.4}$$

For solving QP-Q in (2.4), we follow the line search algorithm of [64] by relaxing QP-Q to the continuous domain

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \ \mathbf{x}^\top (\mathbf{A} - \lambda \mathbf{M}) \mathbf{x} \ \text{ subject to } \quad \mathbf{x} \in [0,1]^{N \cdot L} \tag{2.5}$$

where $\lambda > 0$ is a sufficiently large regularization parameter.

Let $f(\mathbf{x}) = \mathbf{x}^\top (\mathbf{A} - \lambda \mathbf{M}) \mathbf{x}$ denotes the target function. The algorithm in [64] seeks binary solutions in each step. For a given initial vector $\mathbf{x}_0$ with $f(\mathbf{x}_0) > 0$, it increases f in each iteration until it converges to a MAP assignment $\mathbf{x}^*$. Although the formulation is relaxed the returned solutions $\mathbf{x}^*$ are binary in all experiments in [64] and in all our experiments.

Now we show that a binary solution $\mathbf{x}^*$ implies that all mutex constraints are satisfied, i.e., $(\mathbf{x}^*)^\top (\mathbf{M}) \mathbf{x}^* = 0$. Suppose that this fact is not true, i.e., there exists i with $\mathbf{x}_i^* = 1$ that violates a mutex constraint. Then $(\mathbf{x}^*)^\top \mathbf{M} \mathbf{x}^* \geq 1$. Let λ be equal to the sum of all elements of $\mathbf{A}$. Because then $(\mathbf{x}^*)^\top \mathbf{A} \mathbf{x}^* \leq \lambda$, we obtain

$$f(\mathbf{x}^*) = (\mathbf{x}^*)^\top \mathbf{A} \mathbf{x}^* - \lambda (\mathbf{x}^*)^\top \mathbf{M} \mathbf{x}^* \leq 0.$$

A contradiction, since $f(\mathbf{x}^*) \geq f(\mathbf{x}_{(0)}) > 0$.

In the following two sections, we explain how to compute the affinity matrix A, and estimate the mutex matrix M from training data. In the experimental section we discuss our initialization strategy of selecting initial vectors $\mathbf{x}_0$.

2.3 The affinity matrix A

This section explains how to compute the unary and pairwise potentials organized in the affinity matrix A.

2.3.1 The Unary Potential

Recall that elements of the affinity matrix A encode the unary and pairwise potentials of our CRF (see Sec. 2.2.1).

We specify the unary potential of each label assignment (s_i, l_i) as follows:

$$A((s_i,l_i),(s_i,l_i)) = \begin{cases} P(l_i|F,m), & \text{if m} = 1 \\ P(l_i|F,a,h,pt), & \text{otherwise} \end{cases} \tag{2.6}$$

where F are appearance and geometric features of region s_i used in [33], a is the angle between normal vector of s_i and gravity direction ($[0,\pi]$), h is the estimated absolute height above ground, pt is detected plane type $P(pt|l_i)$ [79] (vertical boundary, horizontal boundary, vertical plane, horizontal plane, plane, non-plane), and the binary variable m indicates if a majority of depth information is missing in s_i. For simplicity, we ignore denotation s_i in the following formulas. Assume these observations are independent from each other, then (2.6) can be further decomposed based on Chain Rule:

$$A((s_i,l_i),(s_i,l_i)) = \begin{cases} P(l_i|F)P(m|l_i), & \text{if m} = 1 \\ P(l_i|F)P(a|l_i)P(h|l_i)P(pt|l_i), & \text{o.w.} \end{cases} \tag{2.7}$$

Probability Estimation: The posterior probability $P(l_i|F)$ is the output of Multi-Class Logistic Regression in [33]. The likelihoods of $P(pt|l_i)$ and $P(m|l_i)$

are estimated directly as corresponding histograms on training dataset. For the estimation of likelihood $P(h|l_i)$, it is worth noting that the absolute height h is different from the relative height in previous works such as [79, 33], where it is defined as the height above the lowest point in the image. Typically, the relative height information becomes misleading when the floor doesn't show up in the image. As shown in the left image of Fig.2.1, the horizontal plane is very close to the lowest point of the 3D scene, but actually it is a counter instead of a floor. To solve this problem, we assume that indoor images are captured by human in a natural way. We firstly extract statistical distribution of absolute camera height h_{cam} and for each object class from a training set. We plot the normalized histogram of absolute camera height of training set in the right image of Fig.2.1. It is observed that it roughly obeys a Gaussian distribution. Since height is continuous, the probability density of object l_i, $f_{l_i}(h)$, is derived by Kernel Density Estimation:

$$f_{l_i}(h) = \frac{1}{nb} \sum_{i=1}^{n} K(\frac{h - h_i}{b}) \tag{2.8}$$

where K is a Gaussian kernel smoother and b is bandwidth. Then the likelihood $P(h|l_i)$ is computed as follows:

$$P(h|l_i) = \int_{\mu_c - h' - 3\sigma_c}^{\mu_c - h' + 3\sigma_c} f_{l_i}(h)\,\mathrm{d}h \tag{2.9}$$

where μ_c and σ_c are mean and variance of absolute camera height respectively, and h' is a relative height difference between object and camera. The likelihood $P(a|l_i)$ is estimated in a similar way.

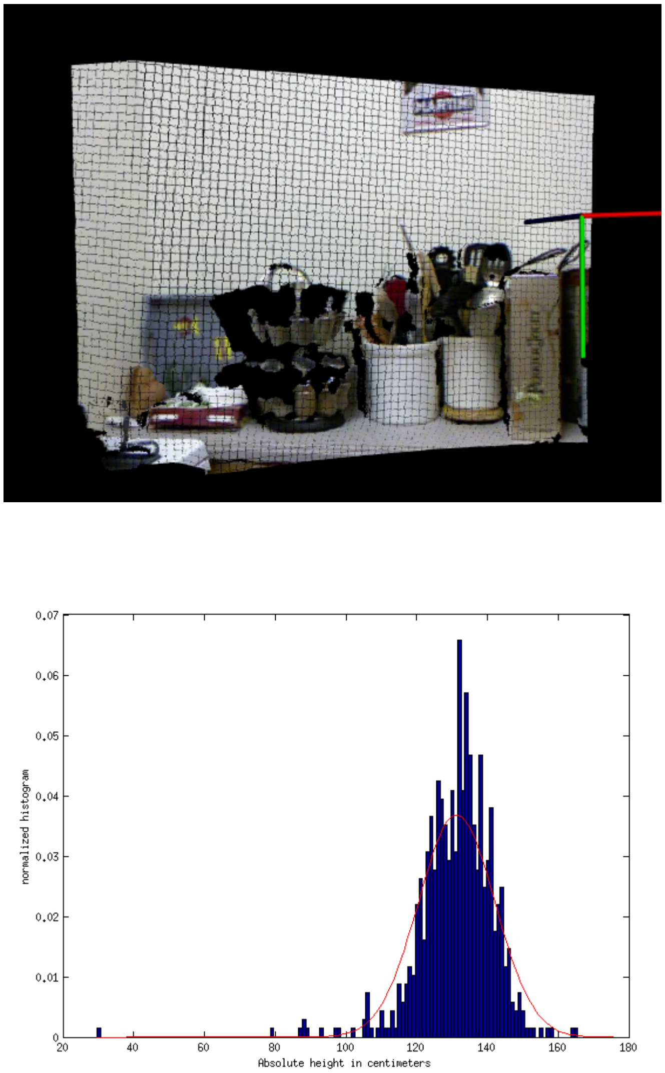


Figure 2.1: Left image: an example of indoor scene (point cloud attached with colors). Camera position and orientation are represented by three orthogonal color sticks. Right image: the normalized histogram of absolute camera height on training set of NYU-V2. The mean value of camera height is around 131 cm.

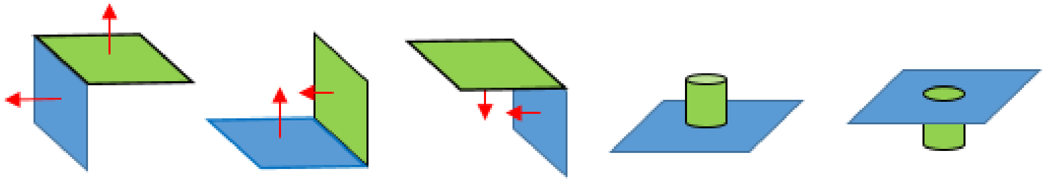


Figure 2.2: Geometric pairwise patterns. Red arrow represents normal vector direction. Blue or green planes indicate that the superpixel is covered by one detected plane structure.

2.3.2 The Pairwise Potential

Further, for all edges in the association graph $\mathbb{G}$, $((s_i, l_i), (s_j, l_j)) \in \mathbb{E}$, we encode the pairwise potentials as the off-diagonal elements of the affinity matrix A. Consider the available 3D geometric information, we define five special pairwise patterns, as is shown in Fig 2.2. While detected edges in 2D image often indicate object boundaries, pairwise patterns imply certain local configurations in 3D space. For example, ”cabinet” and ”counter” usually satisfy the first pattern, while the fourth pattern implies ”table” or ”counter” supports other ”props”.

Co-occurrence Probability Estimation: Except for the five defined patterns above, the other pairwise patterns are considered as one category. We compute adjacency co-occurrence probabilities of the two classes $\Psi^{(k)}(l_i, l_j), k = 1, 2, ...6$ from training data as

$$\Psi^{(k)}(l_i, l_j) = \frac{N^{(k)}(l_i, l_j)}{N^{(k)}(l_i) + N^{(k)}(l_j) - N^{(k)}(l_i, l_j)} \tag{2.10}$$

where $N^{(k)}$ is a function that counts the total number of training images where the event shows up in pattern k. It is worth noting that the first five adjacency co-occurrence probabilities are asymmetric. They also differ from mutex constraints in that the latter captures long-range inconsistency constraints, whereas the former are treated as “soft” constraints that only favors certain pairs of labels at spatially adjacent locations, but in no way strictly prohibit any particular pair.

2.4 Mutex Constraints for Scene Labeling

This section defines the mutex constraints and describes how to estimate them. We use three types of mutex constrains.

Global object co-occurrence constraints encode which objects cannot occur together in a scene. They are called global, because these constraints do not account for a particular spatial layout of co-occurrence. For example, under normal conditions, it is impossible to see both toilet and white board in the same room. In [55], similar co-occurrence constraints are incorporated into the energy function as negative logarithmic potential. Instead, we formulate them as hard constraints using the $(NL) \times (NL)$ binary matrix mutex M_{co} for each pair $v_i = (s_i, l_i)$ and $v_j = (s_j, l_j)$:

$$M_{co}(v_i, v_j) = \begin{cases} 1, & \text{if regions } s_i \text{ and } s_j \text{ with labels} \\ & l_i \text{ and } l_j \text{ never co-exist in a scene} \\ 0, & \text{otherwise} \end{cases} \tag{2.11}$$

Relative height relationship constraints: We observe that relative height relationships typically hold in most indoor scenes. For example, the floor should be lower than chairs, and the ceiling should be higher than pictures. Thanks to Kinect technology, we can easily access depth data for each pixel. Given raw depth data, we align 3D points with gravity direction so that the floor plane lies in $X - Z$ plane, and Y axis represents the height information. The relative height relationship is represented as the $(NL) \times (NL)$ binary matrix M_{rh}:

$$M_{rh}(v_i, v_j) = \begin{cases} 1, & \text{if estimated height relation between} \\ & \text{regions } s_i \text{ and } s_j \text{ contradicts true} \\ & \text{relative locations of objects } l_i \text{ and } l_j. \\ 0, & \text{otherwise} \end{cases} \tag{2.12}$$

Object local support relationship constraints encode basic physical configuration rules of indoor scenes. For instance, counters are usually supported by cabinets, and televisions are supported by dressers. The inverse of these support relations would contradict common-sense knowledge about the real world. We call these constraints local, since they only regulate support relationship between two spatially adjacent regions. In order to evaluate the support relationship of two neighboring regions, we first project 3D points of both regions onto the X-Z plane. If these two projected regions have overlapping area, a support relationship does exist between them. We use a variant of Jaccard Index to measure a ratio of the overlapping area. Let $\alpha(s'_i)$ denote the area of the projected region s_i onto the ground plane. Then, we define the variant of Jaccard Index as

$$\alpha_{ratio}(s'_i, s'_j) = \frac{\alpha(s'_i \bigcap s'_j)}{\min(\alpha(s'_i), \alpha(s'_j))} \tag{2.13}$$

In practice, considering errors from Kinect depth measurement [50] and low level segmentation, we relax the condition to tolerate small overlaps that α_{ratio} is below certain threshold θ. We set $\theta = 0.1$ in all experiments. The support relation constraints are then encoded into the $(NL) \times (NL)$ binary matrix M_{sup}:

$$M_{sup}(v_i, v_j) = \begin{cases} 1, & \text{if } s_i \text{ cannot support } s_j \text{ w.r.t. real} \\ & \text{support relation of objects } l_i \text{ and } l_j \\ 0, & \text{otherwise} \end{cases} \tag{2.14}$$

Generally, we say region s_i can support s_j when the corresponding $\alpha_{ratio} > \theta$, and the centroid height of s_i is lower than that of s_j, given object l_i can support object l_j in the real world.

Finally, the aforementioned three mutex matrices are merged into the unique mutex matrix M as

$$M(v_i, v_j) = M_{co}(v_i, v_j) \vee M_{rh}(v_i, v_j) \vee M_{sup}(v_i, v_j) \tag{2.15}$$

To merge the set of linear constraints of QP-L in (2.2), we set all matrix elements $M((s_i, l_i), (s_i, l_i')) = 1$, if $l_i \neq l_i'$.

Mutex constrains learning: Denote a pair of nodes as $v_i = (si, li)$ and $v_j = (sj, lj)$. We make the assumption that the training set is sufficiently large. For global object co-occurrence constraints, if object class l_i and l_j have been observed present together in at least one training image, then $M_{co}(v_i, v_j) = 0$, otherwise $M_{co}(v_i, v_j) = 1$.

For relative height constraints, we use two auxiliary matrices M_{auxH} and M_{auxL} obtained from training images to encode height relationship rules w.r.t highest point and lowest point respectively. For example, $M_{auxH}(l_i, l_j) = 1$ means the highest point of class l_i always is higher than that of class l_j, while $M_{auxL}(l_i, l_j) = 1$ indicates the lowest point of class l_i always is lower than that of class l_j. Otherwise, no height relative constraint applies to class pair (l_i, l_j). Therefore, $M_{rh}(v_i, v_j) = 0$ when observed height relationship between node v_i

and v_j does not violate any one of rules encoded in auxiliary matrices, otherwise $M_{rh}(v_i, v_j) = 1$.

For local support constraints, we compute the probability of class l_i support class l_j, $P_s(l_i, l_j)$, as the number of positive instances divided by the total number of spatially adjacent regions assigned with labels l_i and l_j. Here,two regions are spatially adjacent if their shared boundary contains at least one pixel and the minimal 3D distance between point clouds is less than $5cm$. Class l_i can not support class l_j if $P_s(l_i, l_j) < 5\%$.

2.5 Experiments

We evaluate our framework on the New York Univeristy (NYU) Depth dataset (v2) and Princeton University SUN3D dataset [93]. The NYU dataset contains 1449 pairs of aligned RGB and depth images which are captured from 27 different indoor scene categories, such as bedrooms, classrooms, kitchens, furniture stores and so forth. In [79] 894 subclasses were grouped into four super-categories: ground, furniture, props and structure for sematic segmentation. [33] extended the total number of object classes for sematic segmentation task from four to 40 classes. In our experiments we follow the settings in [33]. Since only a small portion of images has been labeled in the SUN3D dataset, we use the officially released eight annotated sequences and extract 65 keyframes that cover the content of sequences as much as possible.

Inference settings: As is described in section 2.3.1 , the number of nodes in the weighted graph is relevant to both over-segmentation and class labels. For some extremely complex scenes, the number of regions in the over-segmentation is around 600. But typically the number is around 140. We sort the unary potentials in decreasing order and choose the first k labels as candidates for each superpixel in graph construction stage. If k is too large, it will increase the computational cost and reduce the chance of selecting a correct label. If k is too small, it has a high probability that correct label is not in the candidate list. In the experiment, we set $k = 5$.

As the solver for finding maximum weight subgraph [64] usually converges to a local optimum, multiple initializations are needed to obtain a better performance. We train a SVM classifier by taking unary potentials as features for predicting confidence of each region and rank regions in decreasing order according to it. Then a weighted sampling mechanism is adopted to select a triple of regions as initializations each time. In other words, we set $\mathbf{x}^{(0)}(i) = 1$ if region v_i is selected as one of the three initialization regions. Otherwise, $\mathbf{x}^{(0)}(i) = 0$. Start from $\mathbf{x}^{(0)}$, we obtain a subgraph denoted by the indicator vector $\mathbf{x}^{*}$. In order to enforce the final solution always satisfies the mutex constraints $\mathbf{x}^{\top} M \mathbf{x} = 0$, the parameter λ is set to 1000. We compute x^{*} in (2.5) t times and select the one with highest energy score as the best solution according to $f(\mathbf{x}^{*}) = \mathbf{x}^{*\top} \mathbf{A} \mathbf{x}^{*}$. In our experiment, t is set to 1000.

Performance on NYU dataset: We present both qualitative and quantitative evaluation of our semantic segmentation algorithm. In order to compare our performance directly with the state of the art results in [33, 32, 34], we use the same three metrics: pixel frequency weighted average Jaccard Index, average Jaccard Index and pixel accuracy. We present the quantitative evaluation results in Table. 2.1. We list the best labeling result from [33, 32, 34] in the first three rows of the table respectively. [32] is a journal version of [33]. [34] improved the performance of [33] by using object detections to compute additional features for superpixels. The last row contains labeling results of our inference with mutex constraints. We achieve the best performance in the 40-class segmentation task. In particular, we outperform [33] by 3.4% (fwavacc), 5.4% (avacc) and 5.9%

(pixacc), and outperform [34] by 1.5% (fwavacc), 3.1% (avacc) and 3.5% (pixacc).

In order to demonstrate the effectiveness of mutex constraints, we list the corresponding labeling results obtained by removing mutex constraints from our CRF model in the forth row. In addition, we replace our unary potential in (2.6) with the output of multi-class logistic regression from [33] while keeping the rest of our model unchanged. As shown in fifth row, the performance is slightly worse than our best performance. It indicates that the proposed unary potential formulation in Sec. 2.3.1 is useful for the CRF inference.

Performance on SUN3D dataset: It is worth noting that all 65 images are only used as test set, since we used the system trained on the NYU dataset. In other words, all the parameters and classifiers are exactly the same as those used in the NYU dataset. As only 33 classes are present in the labeled images based on the definition of 40 classes task above, after we obtain the semantic segmentation results for original 40 classes, we project unseen 7 labels into 33 classes. ”floor mat” merges to ”floor” class, ”dresser” merges to ”other furniture” and the other five merge to ”other props”. As is shown in Table.2.2, our model outperforms [33] by 2.8% (fwavacc), 3.4% (avacc) and 5.6% (pixacc). This results clearly demonstrate the generalization power of the proposed model with mutex constraints. We can observe that there are several zero terms in Table.2.2. This might because of the difference in variance of object instance appearances between training set in NYU dataset and SUN3D dataset.

We study the impact of each of our three classes of mutex constraints on the performance of our proposed system in Table 4.2. As can be seen all the constraints contribute to the performance. The most significant mutex constraints are co-occurrence followed by relative height.

Finally we provide some qualitative examples to demonstrate the effectiveness

	wall	floor	cabinet	bed	chair	sofa	table	door	window	bookshelf	picture	counter	blinds	desk	shelves
[33]	**68.2**	81.3	46.2	57.1	36.9	41.2	25.9	14.4	33.5	18.5	42.1	51.5	41.9	5.8	4.4
[32]	67.9	**81.5**	45.0	60.1	41.3	47.6	29.5	12.9	34.8	18.1	40.7	51.7	41.2	6.7	5.2
[34] (R-CNN)	68.0	81.3	44.9	65.0	**47.9**	47.9	29.9	**20.3**	32.6	18.1	40.3	51.3	42.0	11.3	3.5
Ours (noMutex)	66.9	81.0	42.9	55.7	33.5	41.2	28.2	14.0	32.9	20.3	41.2	51.2	41.6	6.6	6.2
Ours ([33]+mutex)	65.1	80.4	48.5	65.2	41.9	51.8	35.3	18.8	**35.1**	**33.9**	**49.1**	49.0	**49.6**	**11.5**	**9.6**
Ours (mutex)	65.6	79.2	**51.9**	**66.7**	41.0	**55.7**	**36.5**	**20.3**	33.2	32.6	44.6	**53.6**	49.1	10.8	9.1

	curtain	dresser	pillow	mirror	floormat	clothes	ceiling	books	fridge	tv	paper	towel	showercurtain	box	whiteboard
[33]	28.5	19.6	30.2	21.5	23.4	7.4	**61.2**	7.0	16.1	7.6	15.7	25.8	7.1	**2.1**	11.7
[32]	26.9	25.0	32.8	21.2	30.7	7.7	**61.2**	7.5	11.8	15.8	14.7	20.0	4.2	1.1	10.9
[34] (R-CNN)	29.1	**34.8**	34.4	16.4	28.0	4.7	60.5	6.4	14.5	**31.0**	14.3	16.3	4.2	**2.1**	**14.2**
Ours (noMutex)	29.5	20.0	30.4	21.6	23.4	8.8	61.1	8.1	16.2	9.8	16.7	27.0	9.1	**2.1**	11.2
Ours ([33]+mutex)	44.8	17.1	34.1	**34.8**	31.8	**14.8**	56.9	**13.2**	20.9	9.5	25.7	**32.3**	22.8	**2.1**	1.0
Ours (mutex)	**47.6**	27.6	**42.5**	30.2	**32.7**	12.6	56.7	8.9	**21.6**	19.2	**28.0**	28.6	**22.9**	1.6	1.0

	person	nightstand	toilet	sink	lamp	bathtub	bag	o-struct	o-furni	o-props	**fwavacc**	**avacc**	**pixacc**
[33]	1.4	21.5	45.4	32.5	23.3	32.6	0	8.0	3.9	21.6	45.1	26.1	57.9
[32]	1.4	17.9	48.1	**45.1**	31.1	19.1	0.0	7.6	3.8	22.6	45.9	26.8	58.3
[34] (R-CNN)	0.2	27.2	**55.1**	37.5	**34.8**	**38.2**	**0.2**	7.1	6.1	23.1	47.0	28.4	60.3
Ours (noMutex)	5.7	21.7	47.1	36.5	23.3	32.6	0	7.8	5.4	23.3	44.8	26.6	60.5
Ours ([33]+mutex)	6.0	18.1	50.4	35.0	29.2	28.9	0	9.4	**8.6**	24.9	47.9	30.4	63.1
Ours (mutex)	**9.6**	**30.6**	48.4	41.8	28.1	27.6	0	**9.8**	7.6	**24.5**	**48.5**	**31.5**	**63.8**

Table 2.1: Performance on 40-class semantic segmentation on the NYU-Depth V2 data set. We compare directly with the best results obtained in [33, 34, 32]. The fourth row shows results of our model without mutex constraints. The fifth row shows results of our model with mutex constraints where our unary potential is replaced with the output of multi-class logistic regression in [33]. The last row contains labeling results of our full model.

	wall	floor	cabinet	bed	chair	sofa	table	door	window	bookslf	picture	counter
[33]	**64.8**	89.9	2.8	29.5	45.1	38.3	42.9	**16.5**	21.8	**15.7**	13.9	40.9
Ours (noMutex)	63.7	**90.1**	5.6	42.9	45.8	38.7	50.6	3.5	26.2	12.3	10.5	43.8
Ours ([33]+mutex)	60.9	89.3	14.5	45.1	46.6	**42.3**	64.8	5.7	**36.4**	0.5	11.3	47.7
Ours	61.1	88.8	**19.8**	**46.3**	**51.1**	41.9	**69.7**	9.3	34.9	2.0	**21.8**	**49.4**

	blinds	desk	nightstd	curtain	toilet	pillow	mirror	sink	clothes	ceiling	lamp	fridge
[33]	**21.3**	0.05	56.6	18.5	**56.4**	6.1	0	**81.4**	0	61.0	0.57	21.8
Ours (noMutex)	19.7	0	58.0	12.8	51.3	13.4	0.16	80.1	0	70.5	4.9	27.3
Ours ([33]+mutex)	6.8	**0.08**	59.4	20.1	55.4	14.1	0	70.6	0	93.2	9.9	65.7
Ours	5.2	0	**62.3**	**20.8**	56.0	**16.7**	**0.2**	67.2	0	**93.6**	**11.5**	**65.7**

	tv	bathtub	towel	bag	box	whtbrd	ostruct	ofurn	oprops	**fwavacc**	**avacc**	**pixacc**
[33]	0	23.7	**13.8**	0	0	**21.2**	0	1.6	10.7	48.2	24.8	60.1
Ours (noMutex)	0	**24.3**	11.1	0	0	17.8	0	**3.4**	11.8	48.8	25.5	61.2
Ours ([33]+mutex)	0	20.3	1.0	0	0	15.5	0	2.8	**12.7**	50.1	27.6	64.7
Ours	0	15.3	2.2	0	0	5.6	0	2.3	11.0	**51.0**	**28.2**	**65.7**

Table 2.2: Performance of 33 classes semantic segmentation task on the SUN3D dataset. All 64 images are used as the test set. Note: since [32] and [34] did not report any results on SUN3D, we cannot include them here.

	NYUV2			SUN3D		
	fwavacc	avacc	pixacc	fwavacc	avacc	pixacc
no co-occur	47.2	28.9	61.9	49.4	27.1	64.1
no rel-h	48.0	30.6	63.2	49.8	27.3	64.5
no support	48.4	31.0	63.6	50.9	28.0	65.3
full	48.5	31.5	63.8	51.0	28.2	65.7

Table 2.3: Ablation Study: We remove the different mutex constraints from the full system and study how the performance degrades.

of our CRF inference model with mutex constraints in Fig. 2.3. The region labelings shown in the second column are directly from [33]. It can be observed that some common sense object configuration rules are violated. For example, the counter (row 2, col 2) is fully supported by a door, and the sofa region (row 4, col 2) has been divided into sofa and bed. The labeling of the same scene turns out to be much more reasonable after enforcing mutex constraints during inference. As is shown in the row 2 and column 3 image, the door area is labeled correctly as cabinet and the labelings of other regions are improved too. Also the big sofa region (row 4, col 3) has been correctly recognized after our inference. The last row shows one labeling example from SUN3D dataset.

2.6 Conclusion

We present a novel method for indoor scene semantic segmentation from RGB-Depth images. We effectively utilize available 3D geometric structures of indoor scenes and learn object relationships directly from training set. Our experimental results demonstrate incorporating hard mutex constraints into a soft CRF model can significantly increase the labeling accuracy. The proposed approach outperforms the state of the art methods on very challenging NYU-v2 RGBD dataset and SUN3D dataset for indoor scene semantic segmentation.

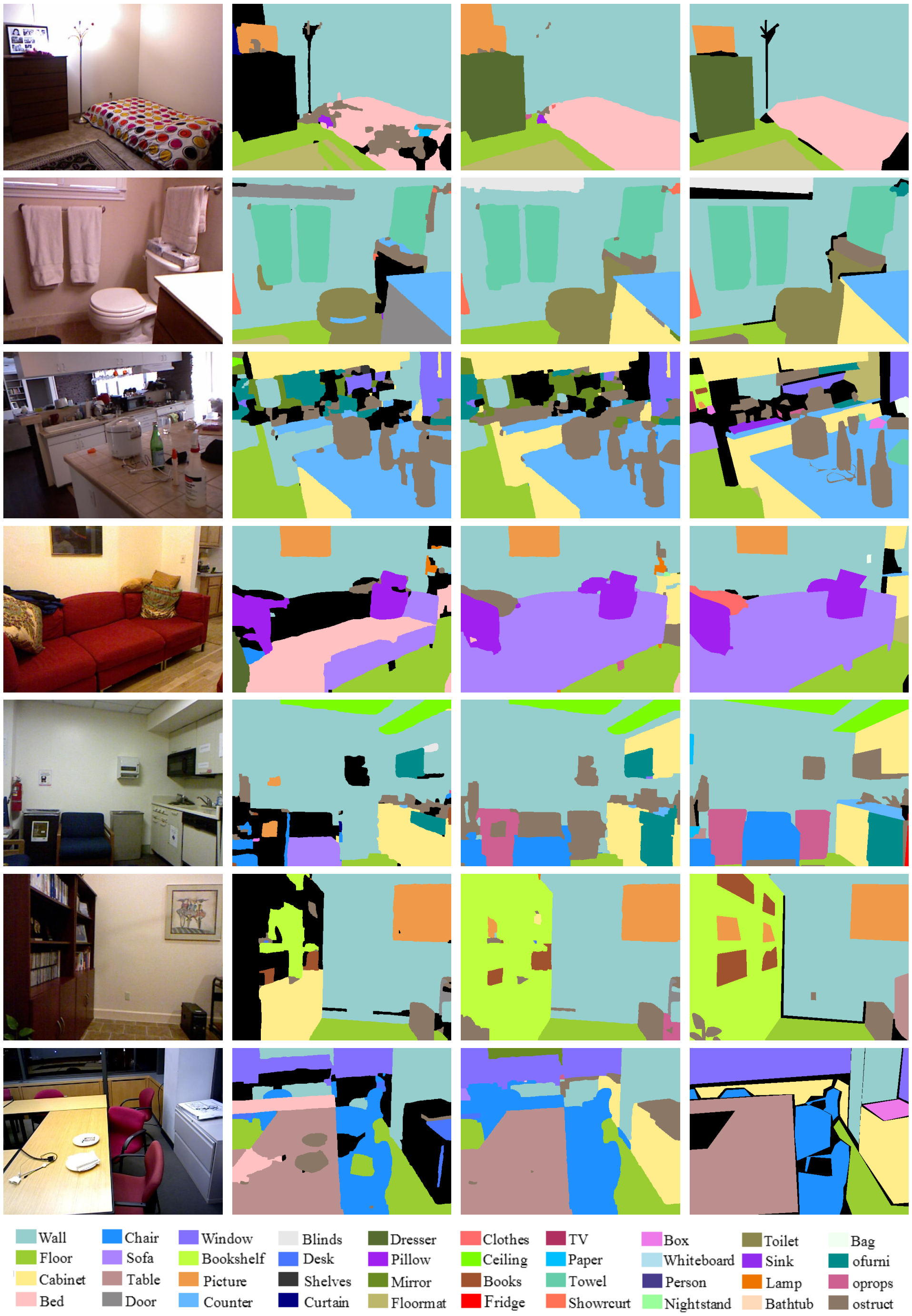


Figure 2.3: Examples of indoor scene semantic segmentation obtained by our system. Column 1 shows the original RGB images, column 2 shows the results from [33], column 3 shows our results after inferring with hard mutex constraints and column 4 shows the ground truth (black areas are unlabeled). Recommend to view in color.

Chapter 3

Unsupervised Object Region Proposals for RGB-D Indoor Scenes

3.1 Introduction

Automatically generating high quality class independent object segmentations is important for many high level computer vision problems such as object detection and recognition. For object recognition, since feature extraction relies directly on the information of its supporting region, the full object region not only conveys global features but also arguably enriches contextual features as confusing background is separated [27]. For object detection, an object can be located at any position and scale in the image. Most of existing work [91, 17] is based on sliding window strategy where exhaustive searching is conducted at various scales and window aspect ratios. However, expensive computation prevents this strategy from utilizing sophisticated feature representations. As an alternative, providing a small set of high quality location hypotheses makes it possible to adopt richer features and complex learning algorithms [27, 37, 15].

Many previous works are dedicated to propose class independent object hypotheses. Uijlings *et al.* [88] proposed a selective search strategy that hierarchically groups similar neighbor superpixels obtained from [18] for predicting object locations. In contrast, besides predicting object bounding boxes, we also aim at providing pixel-level object segments. Carreira *et al.* [8] generated a set of object segments by solving one constrained parametric min-cut (CPMC) problem for each configuration of predefined foreground and background seeds. Lin *et al.* [61] simply extends CPMC by integrating depth for computing potentials. Instead of treating all image region uniformly, we tactically generate hypotheses according to classified regions. Gupta *et al.*[33] generalized gPb-UCM hierarchical segmentation [1] by making effective use of depth information. Arbelaez *et al.* [2] proposed Multiscale Combinatorial Grouping (MCG) to collect segments from multiscale aligned gPb-UCM segmentations. Gupta *et al.* [34] extend MCG

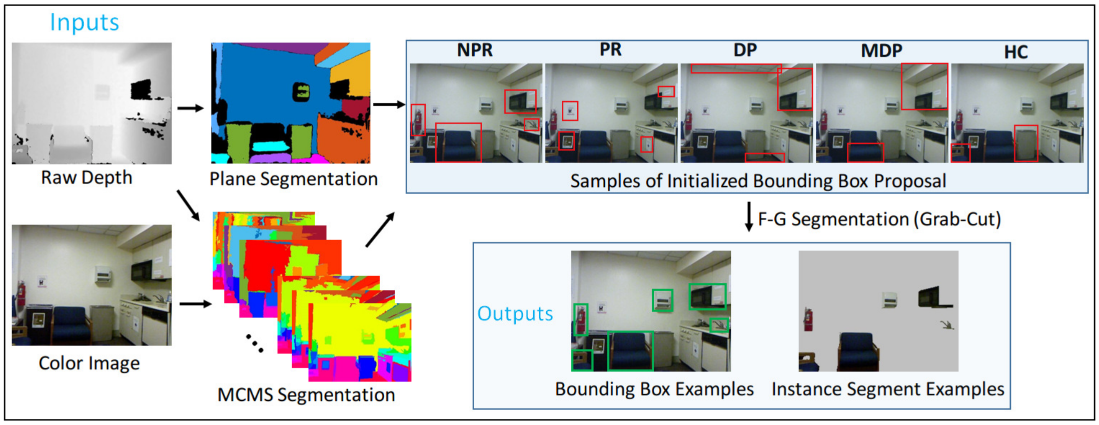


Figure 3.1: The diagram of the proposed system for generating object regions in indoor scenes. Taking one color image and corresponding registered raw depth map from Kinect sensors as inputs, our approach automatically generates object proposals considering five different aspects: Non-planar Regions (NPR), Planar Regions (PR), Detected Planes (DP), Merged Detected Planes (MDP) and Hierarchical Clustering (HC) of 3D point clouds. Object region proposals include both bounding boxes and instance segments. The bottom row shows several examples of generated instances and bounding boxes (green color).

to utilize depth cues for region proposals. While [33, 2, 34] need to learn contour models or/and Pareto front for combinatorial purpose, our approach proposes object regions in an unsupervised way.

We have designed and implemented an integrated system for automatically proposing both object bounding boxes and pixel-level segments in RGB-D images. All the object candidates are generated without any training stage. The overall architecture is presented in the diagram shown in Figure 3.1. The source code of this work will be available online.

We first estimate a general scene layout by fitting planes to 3D points recovered from depth maps. Hence we utilize a common strategy of distinguishing clutter regions from planar regions. In contrast to earlier works like Hedau *et al.* [38], we do not make any assumptions that edges representing joints of walls/floor/celling are visible. Such assumptions were necessary when only RGB data is given. Since we also utilize depth data, the planar surface may represent different objects like table top or other furniture tops. Then we classify planar regions into boundary and non-boundary planes, where a boundary plane is a plane with no objects behind it, e.g., walls and floors. Depending on the scene a table top can also be a boundary plane. Crude bounding box (BB) object proposals are obtained by fitting BBs to planar regions and to segments obtained from Multi-Channel Multi-Scale (MCMS) segmentations and 3D point cloud clustering with the guidance of the estimated scene layout. Finally, we utilize GrabCut [73] to generate segment proposals and refined BB proposals. GrabCut is an excellent foreground object segmenter that is able to dynamically model global object and background properties. However, it has two major limitations. It was developed as (1) interactive human in the loop approach, and it is based on the assumption that (2) the input image contains only one salient object and its background. We address both limitations in the proposed framework and turn GrabCut into a fully automatic, unsupervised segmenter. A general outline of the proposed approach is as follows:

1. Estimate scene layout (Section 3.2.2)
   (a) fitting planes to reconstructed 3D points
   (b) classify planar regions into boundary and non-boundary planes
2. Generate crude BB object proposals (Section 3.2.3)
   (a) Multi-Channel Multi-Scale (MCMS) segmentations
   (b) Euclidean point cloud clustering
   (c) five strategies to generate crude BB proposals
3. Use extended GrabCut to generate segment proposals and refined BB proposals (Section 3.2.1)

We evaluate the proposed approach on standard NYU-v2 RGBD dataset [79] and recent released large scale SUN RGBD dataset [81] in Section 3.3.

To summarize, the main contributions of our approach are: 1) A novel scene structure guided framework for generating bottom-up object region candidates in cluttered indoor scenes. The framework is completely unsupervised, so there is no need to access ground truth information for region proposals, and no bias resulting from the selection of training data. 2) The number of proposed object regions is much less than the state-of-the-arts while the performance is comparable. Hence the proposed framework has a great potential for high-level computer vision tasks such as object detection and recognition. 3) A novel 3D plane segmentation algorithm that is able to detect and segment predominant planar structures of indoor scenes. It is demonstrated to be robust to noise in structured light and other depth sensors.

3.2 Object region proposals in RGBD images

3.2.1 GrabCut Extension

In this section we describe our extension of GrabCut that generates final object segments and BB proposals. The input are initial crude BBs generated by component two.

GrabCut [73] is an iterative GraphCut [4] based segmentation algorithm. Given a region of interest (ROI) in an image, pixels inside ROI are initially labeled as ”unknown” and outside are labeled as ”background”. The goal of GrabCut is to identify the object pixels within this ”unknown” region. In general, two Gaussian Mixture Models (GMMs) of K components ($K = 5$ typically) are used to model foreground and background color distributions, respectively. Model parameters π, μ, Σ are weights, mean and covariance matrices of the $2K$

Gaussian components:

$$\underline{\theta} = \{\pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k), \alpha = 0, 1, k = 1...K\}, \tag{3.1}$$

where α represents the foreground or background. A Gibbs energy function E is defined on the graph G in Eq. (3.2), where each pixel is taken as a node.

$$E = \sum_{i=1}^{n} D(p_i, \alpha, \underline{\theta}) + \sum_{(u,v)\in C} \gamma * [\alpha_u \neq \alpha_v] * exp(-\beta \|p_u - p_v\|^2) \tag{3.2}$$

The data term D encodes the probability of pixel p_i belonging to foreground or background. It is defined as GMM of K components. The smoothness term encourages regional coherence when pixels have similar properties. γ is a constant for balancing data term and smoothness term. C represents the set of pairs of adjacent pixels (we use 4-connectivity), and the constant β is set as inverse of expectation of pixel differences over C defined in Eq. (3.3). At each iteration, the optimal label assignment is obtained by minimizing energy E using GraphCut. Then GMMs parameters in Eq. (3.1) are updated according to the label assignment.

$$\beta = \frac{\sum_{(u,v)\in C} 1}{2\sum_{(u,v)\in C}(\sqrt{\|p_u - p_v\|^2})} \tag{3.3}$$

GrabCut is an interactive segmentation algorithm in that it needs human to provide some hint such as a bounding box around the object candidate. Moreover, it is designed for images consisting of one single salient object with nearly uniform background, e.g., see Figure 3.2.

We observe that when GrabCut is initialized with BBs around object proposals both requirements are met. Our initial guess for object locations is obtained as crude image segments described in Section 3.2.3. Therefore, we initialize it

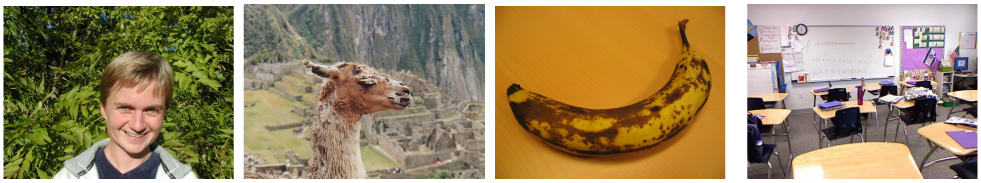


Figure 3.2: Image samples comparison. The first three images are from GrabCut dataset. The last one from NYU-V2 dataset presents a typical cluttered indoor scene.

with BBs around crude segments. In order to increase the chance to cover the whole object by the BB region, we in practice slightly enlarge the BB region. The initial foreground object model is then estimated on the BB region while the initial background model is estimated on the remaining part of the image. It is worth noting that while the whole image is needed for foreground and background model estimations, the object segments are only based on local solution to Eq. (3.2), i.e., the nodes of graph G are pixels within this region. By solving Eq. (3.2) locally for each proposal BB we convert GrabCut into a fully automatic, multiple object segmenter.

Although original GrabCut algorithm shows good performance on foreground segmentation, it often fails to segment objects which have similar color distributions as background, or sometimes decomposes objects into several separated components in image plane. For example, in Figure 3.3, the foreground derived from GrabCut consists of several disconnected pieces and some parts that should belong to the toilet instance are missing.

In order to avoid assigning different labels to pixels that are spatially close, we extend GrabCut by utilizing depth information. We first fill missing data in raw depth map using colorization scheme of [60] and extract 3D points (x, y, z). Then 3D point coordinates (in cm unit) are simply concatenated with RGB channels at each pixel. Hence we consider 6 dimensional GMMs.

Although on average the extended GC3D outperforms the original one due to

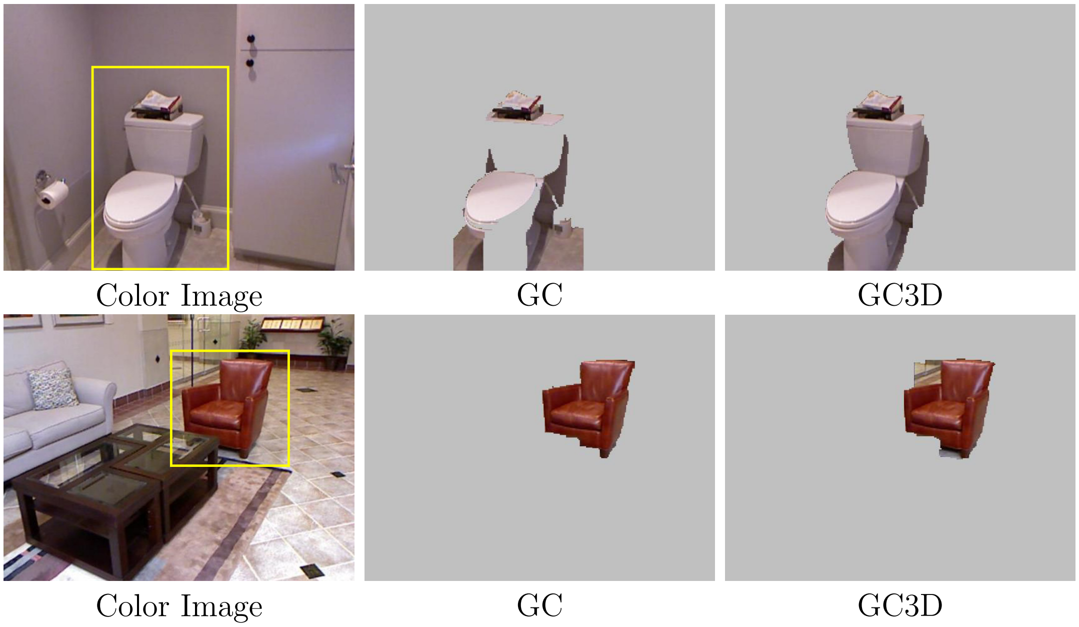

Color Image
GC
GC3D
Color Image
GC
GC3D

Figure 3.3: Examples for foreground segmentation comparison between GrabCut (GC) and its 3D extension (GC3D) both initialized with BBs in yellow frames.

utilization of depth data, e.g., as is shown in Figure 3.3, the toilet instance has been segmented well even if it has similar color distribution to the background, the performance of GC3D may degrade when noise in depth is present. One example is shown in the right scene of Figure 3.3, where a small piece of background is mis-classified. In this case the original GrabCut works well, since the color of the foreground object differs significantly from the background. Therefore, we output the segments from both GrabCut and GC3D as our final segment candidates.

3.2.2 Scene Layout Estimation

Structured indoor environments are often filled with man-made structures and objects, which can be approximately represented with planar segments. We first focus on extracting predominant planar regions such as wall, floor, blackboard, cabinet etc from dense point clouds derived from the depth image, not only because planar regions themselves are meaningful but also they are helpful for generating object hypotheses by focusing on point cloud not explained by major planes. As is well known, comparing to laser range finder, depth information from Kinect and similar sensors has low depth resolution and a limited distance range. To deal with such kind of noise contained in the depth image, traditional plane segmentation methods [79, 49] resort to appearance based cues from RGB image. For example, [79] infers the assignment of points to planes by modeling Graph-Cuts with color and depth information, while [49] utilizes detected line segments in color image to decide about region continuity. However, we believe that integrating color information here is a double sword, since the RGB image maybe noisy. Therefore, we use only 3D point clouds for plane detection and propose a plane segmentation algorithm that is designed to work with point clouds generated by Kinect like sensors.

Plane Segmentation: We first determine the direction of gravity [33] and then rotate the point clouds to make them aligned with room coordinates. A normal vector N_p is estimated for each point p that has valid depth information, which we call a *valid* point. To initialize plane candidates, we uniformly sample triple point sets on the depth map and store them in set $T = \{(p_{i1}, p_{i2}, p_{i3}), i = 1, 2, ...\}$. Then for each $t_i \in T$ we find inliers S_i in the 3D space and a plane candidate P_i in RANSAC framework [20]. Each inlier is represented by a pixel in the depth map and a corresponding 3D valid point. See steps 1-6 in Algorithm 1. The definition of inliers follows below.

In general, a point is considered as an inlier when its distance to the plane is within certain constant range [36, 69]. However, as indicated in [51], depth resolution (i.e., minimum depth difference that can be measured by a sensor) is inversely proportional to the depth, which is defined in Eq (3.4), where f is focal length, b is base length of Kinect sensors, m is the parameter of a linear normalization and Z represents depth value. Therefore, we vary plane inlier distance tolerance based on depth resolution rather than heuristically choosing one constant threshold.

$$D_{tol} = (\frac{m}{fb}) * Z^2 \tag{3.4}$$

We define a point p to be an *inlier* of plane P_i if $d(p, P_i) < D_{tol}(d)$, where d is the Euclidean distance in 3D space. We then remove plane candidates which have small number of pixels and merge spatially close and nearly coplanar planes. However, many fake planes which consider points of other non-planar objects as inliers exist due to noisy surface normals and depth. To filter out fake planes (steps 10-20 of Algorithm 1), we first compute connected components $CC_i = \{c_{i1}, \cdots, c_{ij}, \cdots\}$ of pixels in S_i in the depth map. Then we fit a plane $P_{c_{ij}}$ to 3D points in each connected component c_{ij} and estimate the plane parameters, including its normal $N_{P_{c_{ij}}}$. We assume that $N_{P_{c_{ij}}}$ should be at least similar to N_{P_i}. Hence, if the angle between N_{P_i} and $N_{P_{c_{ij}}}$ is large, we remove the connected component c_{ij} from CC_i. We then re-estimate plane parameters of P_i based on inlier points in survived components.

For plane segmentation, which is performed on the depth map, we assign to plane P_i corresponding pixels in image plane if $d(p, P_i) < 3D_{tol}(p)$. The goal is to avoid artificial holes on plane segments on the depth map. Since now preliminary plane segments are available, we further remove false positive plane segments by checking statistical features, i.e., average to-plane distance d_{avg} and average normal angle $angle_{avg}$ between the average of normals of points in S_i and plane normal N_{P_i}. More details are illustrated in Algorithm 1. To our best knowledge, we are the first to segment multiple indoor planes by considering quadratic sensor noise model and relying purely on 3D point cloud. [51] only proposed the depth noise model but did not apply it to multiple plane segmentation. [79] use a linear noise model to detect planes and use color information for pixel assignment.

Plane Classification: After major planar regions are detected, we further classify them into boundary and non-boundary planes, where a boundary plane is a plane with no objects behind it. Supposed that the normal vector of a plane points towards the viewer, we compute the ratio r of points on the other side of the plane to the total number of points in the room. Ideally, a planer region is a boundary plane if r is zero. We set r to 0.01 to tolerate the sensor noise.

Algorithm 1 Plane Segmentation of Indoor scenes

```
Require: Raw depth map and its 3d point cloud $\{p_i, i = 1, 2, ...\}$ in room coordinate system.
Ensure: A series of major plane segments.
compute distance tolerance $D_{tol}$ according to Eq. 3.4 and normal vector $N_p$ for each valid point.
uniformly sample triple point sets $T$ on image grid.
for $t \in T$ do
    get plane candidate $P_i$ and its inlier set $S_i = \{p | d(p, P_i) < D_{tol}(p), \langle N_{P_i}, N_p \rangle < th_N\}$
    discard plane $P_i$ if the inlier number in $S_i$ is less than $th_{min\_pts}$.
end for
sort $\{P_i\}$ w.r.t # of inliers in decreasing order and remove heavily overlapping ones.
merge spatially close and nearly parallel plane candidates.
remove points that have multiple plane IDs from sets $\{S_i\}$.
for each survived $P_i$ do
    compute connected components $CC_i = \{c_{i1}, \cdots, c_{ij}, \cdots\}$ of $S_i$ in the depth map.
    for each component $c_{ij}$ do
        remove $c_{ij}$ from $S_i$ if its size is small. O/W, estimate plane $P_{c_{ij}}$ by RANSAC.
        if $acos(N_{P_i}, N_{P_{c_{ij}}}) > 10^\circ$ then
            add new plane $P_{c_{ij}}$ to the plane set if $size(c_{ij}) > th_{min\_pts}$;remove points $c_{ij}$ from $S_i$.
        end if
    end for
    discard $P_i$ if the remaining inliners is less than $th_{min\_pts}$.
end for
re-estimate plane parameters for $P_i$ by RANSAC on its current inliers.
re-sort planes w.r.t their number of inliers in descreasing order.
assign pixels to planes one by one if $d(p, P_i) < 3 \cdot D_{tol}(p)$ and $\langle N_{P_i}, N_p \rangle < th_N$
for each plane, remove its components where $d_{avg} > D_{tol_{avg}}$ and $angle_{avg} > th_a$
filter out plane component whose size is less than $th_{min\_pts}$.
```

3.2.3 Initial crude region and BBs proposals

Indoor scenes are usually composed of several predominant planar geometric structures such as ceiling, floor, wall, cabinet, etc and many small cluttered things including clothes, bottles, cups, etc. Based on this prior knowledge, we propose to generate object regions by different strategies with respect to the geometric properties of image regions, rather than treating all image regions uniformly. Since low level image segmentations often indicate cues for object candidate shapes and locations, we adopt Multi-Channel Multi-Scale (MCMS) segmentations for obtaining crude object segments. Note that segments obtained from MCMS segmentation are crude (either too coarse or too fine), and they do not represent final instance segments we are looking for. We utilize five different strategies, described below, to select crude segments for initializing object BB proposals.

For objects in Non-Planar Regions (NPRs) (e.g., cups, faucets etc.) all segments except those that have small overlapping area with NPRs are used, while for objects in Planar-Regions (PRs) (e.g., pictures, papers, etc.) only segments that are generated from RGB channel segmentation and lie in the planar areas are reserved. Segments from Detected Planes (DPs) can be used directly for objects such as ceiling, wall, floor, etc. However, sometimes big objects are inclined to be decomposed into several planar regions (e.g., bed and sofa), and then it is very likely that the proposed bounding boxes are not covering the whole object.

To address this problem, we focus our attention on non-boundary planes, which usually represent big objects like bed or other furniture. For each non-boundary planar region, we then find its border points, which are used to compute minimum distance to other non-boundary planer regions. This distance is used to merge non-boundary planar regions that are close in 3D space (within 5cm) to obtain Merged Planar Regions (MPRs). BBs are then fitted to MPRs.

In addition, we apply Hierarchical Clustering (HC) to 3D point cloud to obtain object instances that are ambiguous in the color image while separated well in 3D world.

3.2.3.1 Multi-Channel Multi-Scale (MCMS) image segmentations

Indoor scenes typically consists of a relatively large number of alike objects that are often cluttered and in disorder, which makes our task of finding a small set of high quality class independent object candidates non-trivial. Moreover, the contents in images are intrinsically organized in a hierarchical way. For example, in Figure 3.5, the "bed" can refer only to mattress and box part or include everything on its top such as sheet and pillows. Besides, indoor scene objects are always in different sizes, colors and shapes, and presented under various light conditions. Therefore, it seems impossible to get object partitions from a generic segmentation strategy that relies on a single signal. Based on these observations, we propose to initialize object locations by using low level segmentations from multiple signal channels and image scales.

In this paper, we get low level segments based on two unsupervised segmentation methods: graph based segmentation (GBS) [18] and watershed based segmentation (WBS) [67] for their high computing efficiency, but other excellent generic image segmenters such as gPb-UCM [1] could also be used in our framework. For GBS, except for using color image alone, we use depth map and combined RGB-D channels for computing the edge weights of neighboring pixels at different scales respectively. To be more specific, in total we collect superpixels from 10 different layers based on GBS including 4 scales from color channel, 3 scales from depth channel and 3 scales from RGB-D fusion channels. In RGB-D fusion channels, we normalize associated 3D point coordinates extracted from raw depth into $[0, 255]$, and compute affinity weights as the maximum gradient value of RGB and depth channels. In practice, the segmentations from multi-scale GBS are helpful for finding most of object locations but are inclined to ignore some salient objects that only occupy small number of pixels in images. To fixed the problem, we adopt WBS as a complementary segmentation tool, which shows more respect to salient object boundaries.

In WBS, we first smooth input maps using a 9×9 Gaussian mask and then compute gradient magnitude maps. Since we care more about strong boundaries, we normalize gradient maps into $[0, 1]$ range and keep values that are above a predefined threshold (we use 0.1 in this paper). This is also useful for avoiding generating segments that are too fine. Then we apply watershed algorithm to gradient maps estimated from intensity image in CIELAB color space, rawDepth map, inpainted depth map, and normals map, respectively. For each gradient map, we obtain one single layer segmentation. As is mentioned in Section 3.2.3, using superpixels from color channel GBS only for object proposal in planar regions is an effective strategy for reducing redundant proposals obtained from other signals. But we do not apply the same strategy to WBS segmentations.

3.2.3.2 Euclidean clustering of point cloud

The goal of point cloud clustering is to partition 3D points into several meaningful structures. Taking advantage of 3D geometry of 3D scenes, it is able to remove ambiguities between object instances caused by similar colors or poor illuminations in indoor environments. Take the two chair instances that are within the yellow bounding box in Figure 3.4, for example. While it is very difficult to distinguish them based on color image alone, they are well separated in the 3D world. We adapt the Euclidean clustering algorithm in [75] for generating object candidates from 3D points.

We first remove detected predominant planes (both horizontal and vertical)

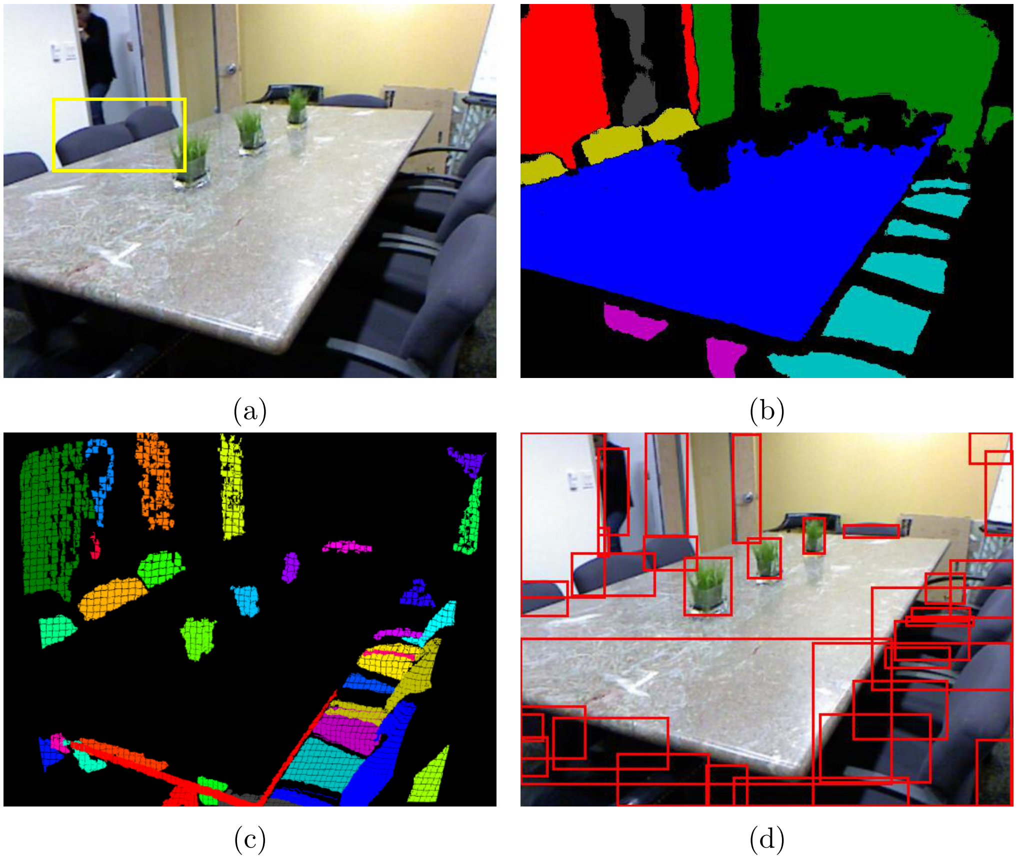


Figure 3.4: An example of Euclidean clustering of 3D point cloud. (a) Color image: two adjacent blue chair instances within yellow bounding box share similar appearance. (b) The plane segmentation (refer to section 3.2.2). (c) 3D point clusters at 5 cm scale. (d) Proposed bounding boxes (red) based on point clusters.

from point cloud before clustering. Then we create a Kd-tree representation for the remaining 3d points. As depth data from Kinect sensor are noisy, we filter out sparsely distributed or isolated points (less than 30 points within $1cm^3$) and get a point cloud P. Starting from any point $p_i \in P$ as one cluster, we search for its unlabeled neighbors that are within certain radius d_{th} and add them into the cluster. Then we keep searching neighbors for each member of current cluster until the size of cluster is stable. Clustering terminates when all points in P are assigned a cluster label. Similar to 2D segmentation, we set multiple radii d_{th} for getting multiple scale clusters ($d_{th} \in \{2, 5, 10\}cm$). In Figure 3.4, we present one example of Euclidean clustering in a typical office environment, where both blue chair instances and green plant instances are well identified. Moreover, planar instances such as door and white board are also identified. We use red bounding boxes to mark identified instances.

3.3 Experiments

We compare our method with the state-of-the-art methods on the NYU Depth V2 dataset [79]. Since some of baselines generate their object proposals with supervised learning, for fair comparison, we follow the standard split (i.e., 795 training images / 654 test images), and report results on test set, except for plane segmentation evaluation which is measured on the whole dataset. To demonstrate, the general applicability of our approach, we also test on a large scale dataset ”SUN RGBD” [81] without changing any parameters.

In our approach, we provide two sets of bounding boxes: one called BB-init, which are all bounding boxes used to initialize foreground segmentations (FG) in Section 3.2.1, and the other called BB-full that includes bounding boxes fitted to segments obtained by FG plus bounding boxes fitted to segments obtained by plane and watershed segmentations.

3.3.1 Evaluating Plane Segmentations

We compare with two state of art works [79, 49] with respect to plane segmentation on RGB-D images. For qualitative evaluation, we provide segmentation results under different indoor scenarios in Figure 3.5. Both [79] and [49] utilize color image with depth map for region smoothness consideration. However, they either fail to detect certain predominant planar regions or have planar regions spread across multiple object boundaries, while our method shows more respect to geometric boundaries and have most major planes detected (e.g., the window frame plane in the office). In addition, we provide quantitative evaluation in Table 3.1. Following [49], we consider both Exactly Planar Classes (EPC) (e.g., floor, ceiling, wall, cabinet etc) and Exact and Nearly Planar Classes (E+NPC) (e.g., bookshelf, books, sofa, bed etc) for evaluation. We compare the obtained planar segments with planar object instances by averaged Jaccard Index. In both cases, our method outperforms the other two methods.

Failure cases analysis

In the Figure 3.5, we present 4 scenes that have failure detection cases. One case is false positive. For example, in the fifth row, the man's body and part of his arm has been identified as one plane. And in the 6th row, the surface of the ladder is merged with the green bag since they are co-planar in the space. The other case is missing detection. Taking the 7th row for example, a majority part of scene is lacking of depth data since infra-red light was lost under a strong sun shine. Another example is from last row where the table is transparent so that the raw depth does not reflect a real plane surface.

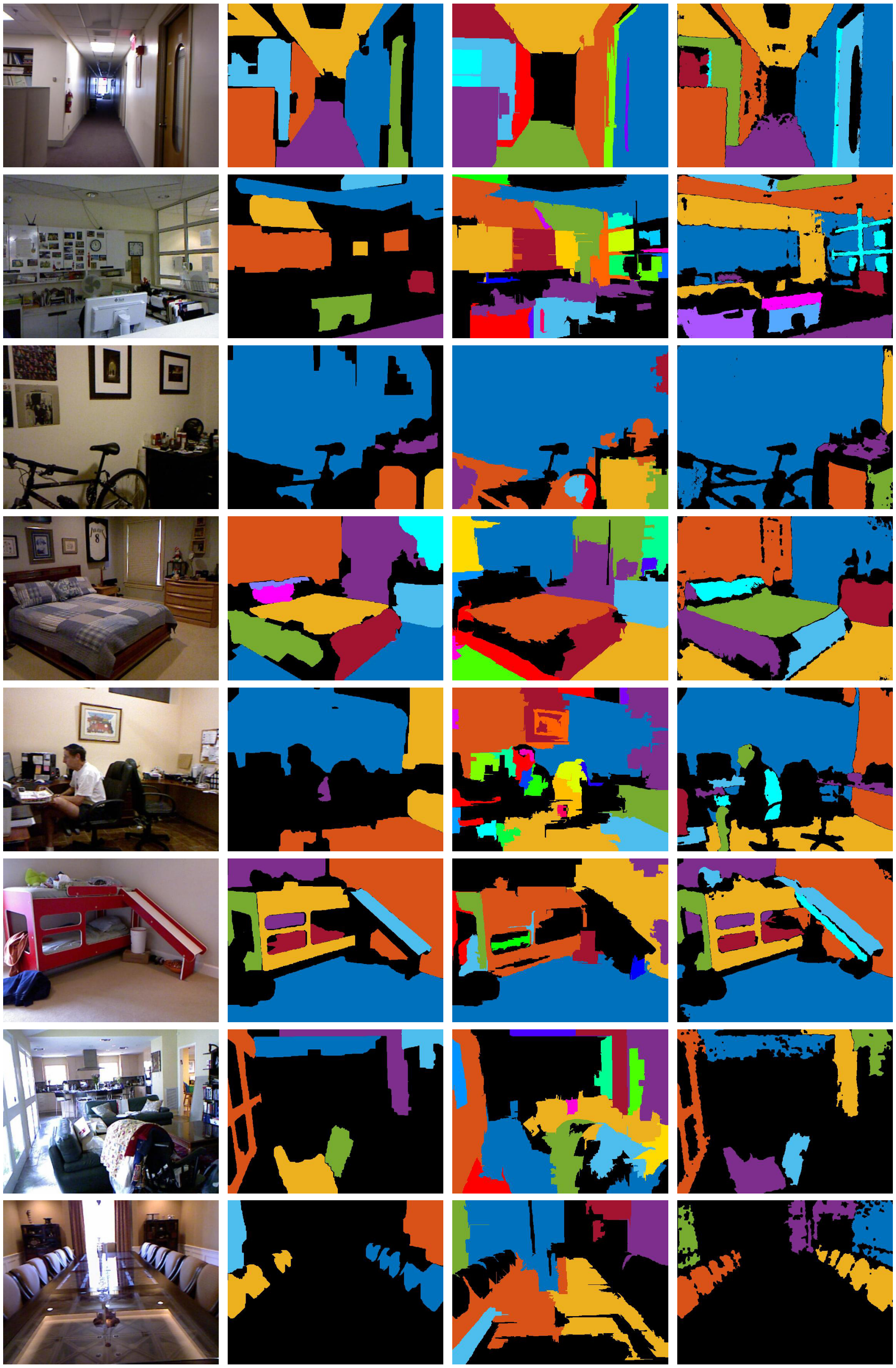


Figure 3.5: Examples of qualitative plane segmentations for RGB-D indoor scenes. The 1st column are original color images. The 2nd column presents plane segmentations by Silberman *et al.* [79]. The 3rd column shows plane segmentations by Khan *et al.* [49]. We present our segmentation results in the last column. The black pixels mark non-planar objects. The last four rows show some failure cases.

Method	Silberman *et al.* [79]	Khan *et al.* [49]	Ours
EPC JI	34.15%	33.87%	**36.72%**
E+NPC JI	30.91%	32.33%	**32.67%**

Table 3.1: Performance comparison of plane segmentations on NYU Depth V2 dataset. Jaccard Index (JI) is used as metric for evaluating obtained planar segments w.r.t. both Exactly Planar Classes (EPC) and Exact and Nearly Planar Classes (E+NPC).

3.3.2 Evaluating Object Region Proposals

3.3.2.1 NYU-V2 Dataset

In this section, we compare our object proposal approach with five state-of-the-art class independent object proposal methods on NYU-V2 RGBD dataset. MCG[2], MCG3D[34], and gPb3D [33] are supervised methods, and CPMC [8], CPMC3D [61] are unsupervised methods (excluding segments ranking). Following MCG [2], for object segmentation evaluation, we compute global Jaccard Index (i.e., intersection over the union of two sets) at instance level as the average best overlap for all the ground truth instances in the dataset, in order to avoid bias on object sizes. For object location proposals, we define bounding box proposal recall score as the ratio of positive predictions that exceed 0.5 Jaccard score, over the number of all ground truth object instance locations. As is shown in Table 3.2 and Figure 3.6, our method achieves the best performance (**91.1%**) for object location proposals while our number of maximum proposals is only **40%** of the rank-2 method MCG3D[34]. Moreover, our initial bounding boxes require even less proposals (**21%** of [34]) while the recall score only degrades 2% w.r.t the best performance.

For object instances proposal, our method also show very competitive performance: our score is 0.9% less than the best performance but our number of proposals is less than half of theirs. It is worth noting that we do not rank our

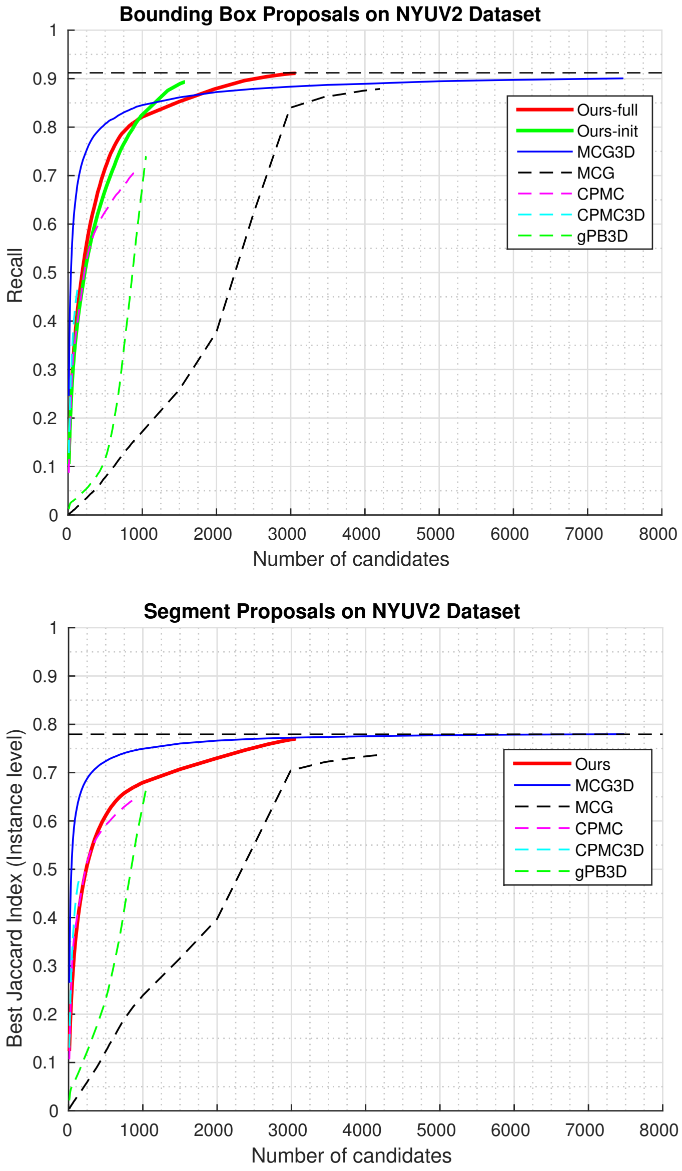


Figure 3.6: Quantitative evaluation of object region proposals with respect to the number of object candidates on NYU-V2 RGBD dataset. Left: recall curves on proposed bounding boxes evaluation. Right: average best Jaccard Index curves on proposed segments evaluation. Note the curves of MCG3D and CPMC3D are based on supervised ranking of segments, while the other curves including ours do not use any ranking.

	[33]	[8]	[61]	[2]	[34]	Ours-BB-init	Ours-BB-full
Global Best (bbox)	0.74	0.706	0.473	0.879	0.901	0.893	**0.911**
Global Best (seg)	0.67	0.646	0.478	0.737	**0.779**	-	0.77
# Proposals	1051	885	138	4202	7482	1575	3066

Table 3.2: Performance comparison of best global Jaccard Index at instance level for both bounding box and segment proposals on NYU-V2 RGBD dataset.

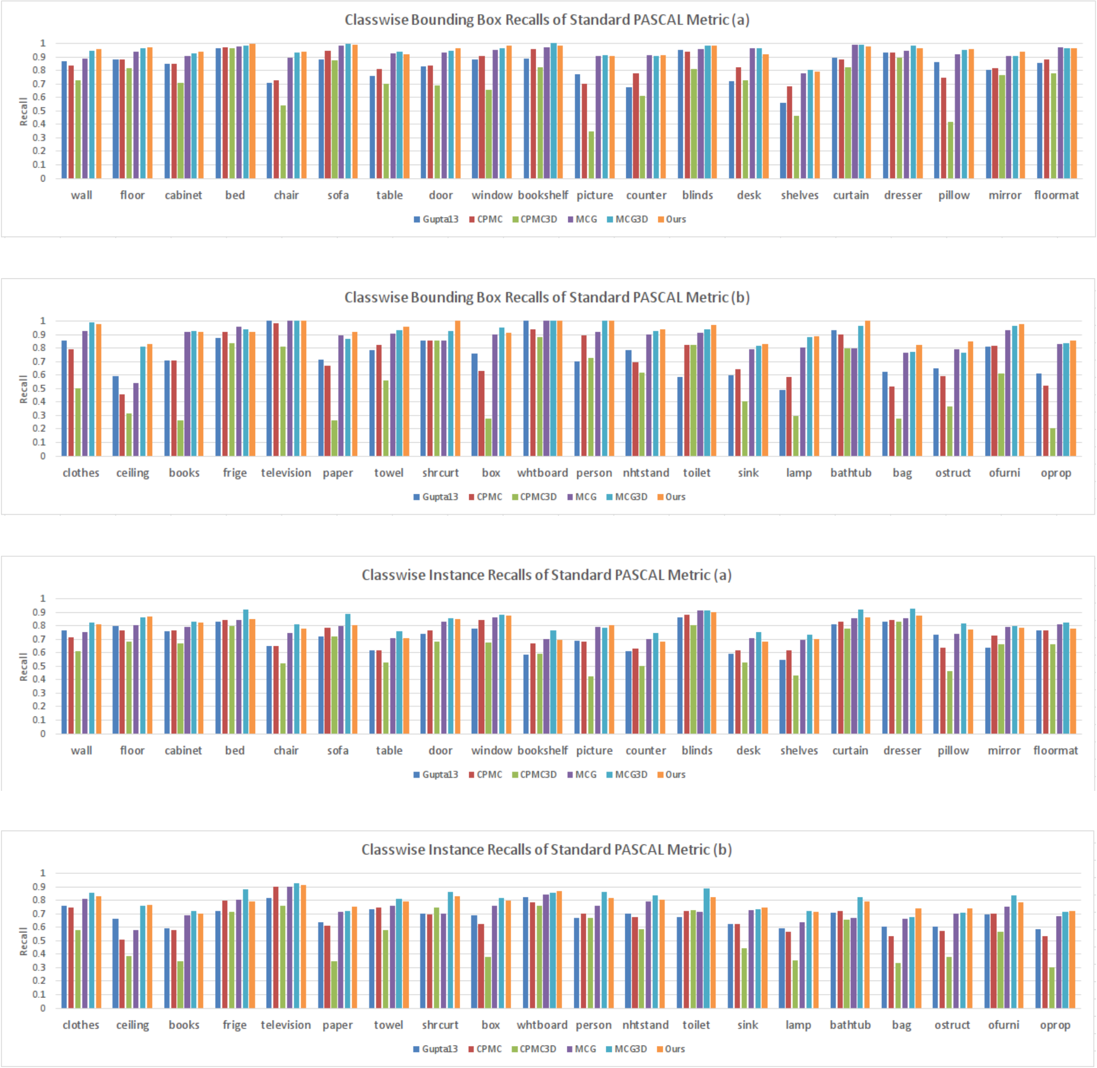
bounding box proposals in our result presentation, while [34, 61] perform supervised ranking. Since we already provide high quality object segmentations with much less number of proposals in a complete unsupervised framework, ranking proposals is beyond the scope of this paper.

In addition, we provide results of global Jaccard index at class level for both object location and segmentation proposals in Figure 3.7. We divide 894 classes into 40 classes following the definition of [33] including 37 specific object classes and 3 abstract classes: ”other struct”, ”other furniture” and ”other props”, which include 68, 82, 707 subclasses respectively. We obtaine best performance on 26 classes for object location proposals and 9 classes for segment proposals. It is worth noting that our method achieves best performances on the three abstract classes for object location proposals. It indicates that our approach is general to different object types since abstract classes cover 95.8% subclasses and 32.3% instances on the test set.

Except for quantitative evaluation, we also provide qualitative evaluation for proposed object regions in Figure 3.8. The first six scenes show objects that have been segmented successfully, and in the last two rows we list several failures cases. The grabcut segmenter is inclined to fail either when the foreground and background have similar color information, or when the foreground object is too small or has irregular shapes (e.g, plants).

Ablation Study

In order to understand the individual impact of the five proposal strategies on


Classwise Bounding Box Recalls of Standard PASCAL Metric (a)

Classwise Bounding Box Recalls of Standard PASCAL Metric (b)

Classwise Instance Recalls of Standard PASCAL Metric (a)

Classwise Instance Recalls of Standard PASCAL Metric (b)

Figure 3.7: Classwise (40-class) performance comparisons based on the standard PASCAL metric (Jaccard Index) at object instance level for both bounding box and segment proposals on the NYU-v2 RGB-D dataset.

	no NPR	no PR	no DP	no HC	no MPR	Ours-full
Global Best (bbox)	0.666	0.813	0.889	0.897	0.901	**0.911**
Global Best (seg)	0.610	0.699	0.733	0.748	0.753	**0.77**

Table 3.3: Ablation study: each time we remove one of the five object proposal strategies from the full system and report how the performance degrades with respect to both bounding box and segment proposals.

the performance of our RGB-Depth object proposal system, we evaluate our algorithm on the NYU-V2 RGB-D dataset by removing one strategy each time. The corresponding results are listed in the table 4.2. As can be seen all the strategies contributes to the performance. The ranking of strategies in decreasing significance order is NPR, PR, DP, HC, and MPR.

3.3.2.2 SUN RGBD Dataset

We also test our unsupervised approach without changing any parameters on the recently released SUN RGBD dataset. SUN RGBD is a large scale indoor scenes dataset with a similar scale as PASCAL VOC. It contains $10,335$ RGB-D images in total, which are collected from four different active sensors: Intel RealSense, Asus Xtion, Microsoft Kinect v1 and v2. While the first three sensors obtain depth map using IR structured light, the Kinect v2 (kv2) estimates the depth based on time-of-flight. With respect to raw depth data quality, kv2 can measure depth with the highest accuracy but at the same time there are a lot of small black holes in the depth map due to light absorption or reflection. The RealSense has the lowest raw depth quality.

	SUN RGB-D dataset [81]				
Sensors	Kinect v1		Kinect v2	RealSense	Xtion
Resources	B3DO [46]	NYUV2	*	*	SUN3D [93]
Global Best (bbox)	0.929	0.911	0.908	0.909	0.912
Global Best (segment)	0.742	0.77	0.746	0.745	0.752
# proposals	2972	3066	2971	4628	2969

Table 3.4: Performance evaluation of our method on the large scale SUN RGB-D dataset, the images of which are collected from four different RGB-D sensors. *: newly captured RGB-D images in [81].

As can be seen in Table 3.4, in general, our approach exhibits similar performance to the NYUV-2 dataset. We observe that while the bounding box predictions show consistent performance, the accuracy of instance proposals de-

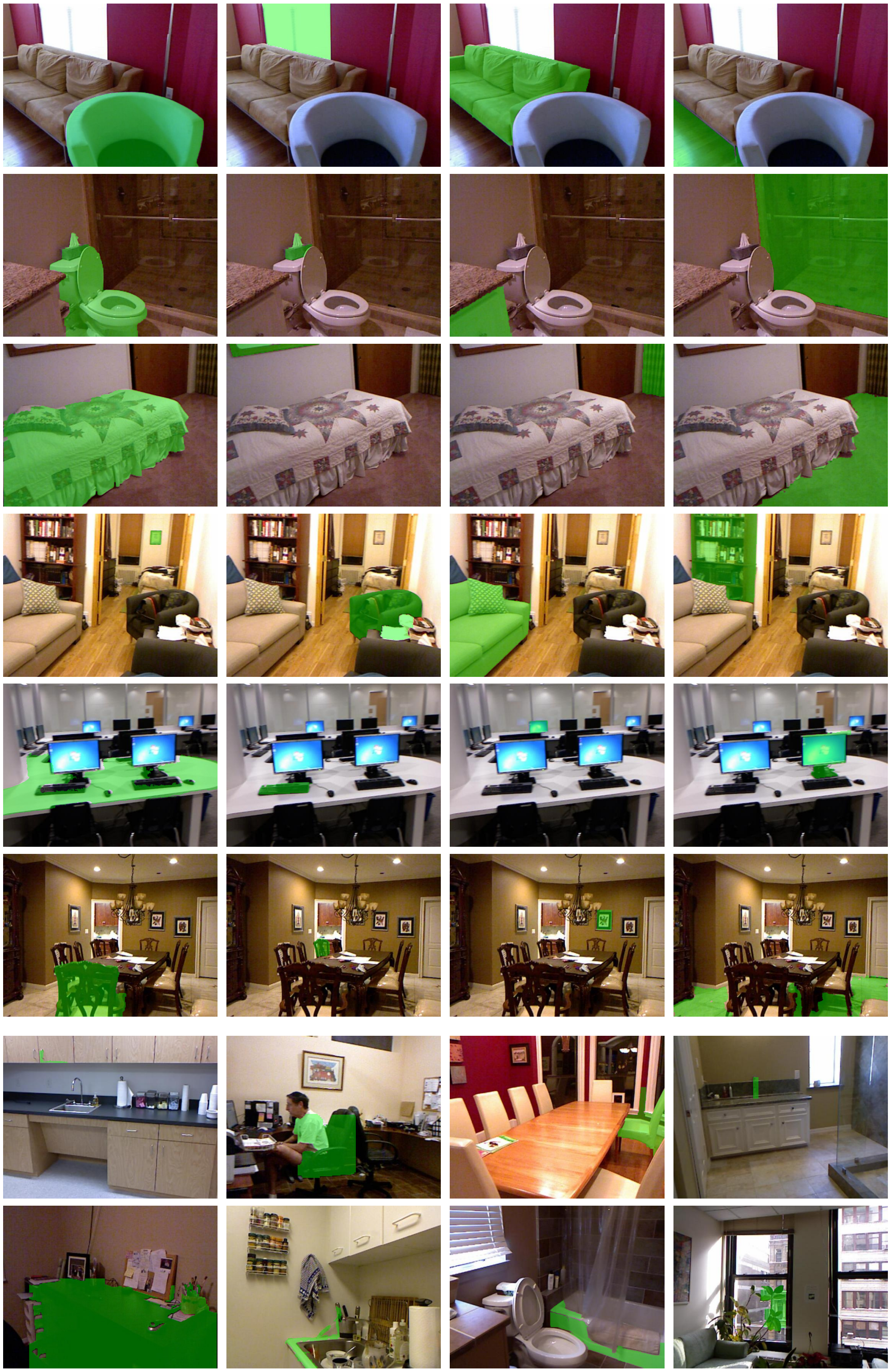


Figure 3.8: Qualitative performance evaluation for proposed object segments on NYU-V2 RGBD dataset. Object proposals are highlighted with green color. And several failure cases are provided at the last two rows.

grades around 2%. This reasonable degradation might be due to higher variance in sensor depth resolution. The average number of proposals is similar to the number on NYUV-2 dataset except for the tests on RealSense data, where it increases by around 50%. This is expected as the effective depth range of RealSense is very short (depth becomes very noisy or missing beyond 3.5m).

3.4 Conclusion

We propose an unsupervised unified framework for class independent object bounding box and segment proposals. Our method produces object regions with very comparable qualities to the state-of-the-arts while requiring much less proposals, which indicates its great potential for high level tasks such as object detection and recognition. The source code is available on authors' websites. (**https://github.com/phoenixnn/RGBD-object-propsal**).

Chapter 4

Amodal Detection of 3D Objects: Inferring 3D Bounding Boxes from 2D Ones in RGB-Depth Images

4.1 Introduction

Object detection is one of the fundamental challenges in computer vision, the task of which is to detect the localizations of all object instances from known classes such as chair, sofa, etc in images. Traditionally, detected object instances are represented by 2D bounding boxes around visible counterparts on images. Although 2D rectangles can roughly indicate where objects are placed on image planes, their true locations and poses in the physical 3D world are difficult to determine due to multiple factors such as occlusions and the uncertainty arising from perspective projections. However, it is very natural for human beings to understand how far objects are from viewers, object poses and their full extents from still images. These kind of features are extremely desirable for many applications such as robotics navigation, grasp estimation, and Augmented Reality (AR) etc. In order to fill the gap, a variety of efforts were made in the past decade including inferring 3D object localizations from monocular imagery [19, 40, 68, 9], and 3D object recognitions on CAD models [92, 85]. But these works either rely on a huge number of ideal 3D graphics models by assuming the locations are known or are inclined to fail in cluttered environments where occlusions are very common while depth orders are uncertain.

The recent advent of Microsoft Kinect and similar sensors alleviated some of these challenges, and thus enabled an exciting new direction of approaches to 3D object detection [62, 52, 35, 31, 63, 82, 83]. Equipped with an active infrared structured light sensor, Kinect is able to provide much more accurate depth locations of objects associated with their visual appearances. The RGB-Depth detection approaches can be roughly categorized into two groups according to the way to formulate feature representations from RGB-Depth images.

In general, 2D approaches start by exploiting proper 2D feature representations on image planes for object detection and building models to convert 2D results to 3D space. While 3D approaches start by putting detection proposals directly in 3D space for extracting features from 3D point cloud within 3D windows. The competition to determine whether 2D or 3D approaches represent the right direction for 3D amodal object detection is super intense: [82] utilized 3D sliding window to directly infer detections in 3D space and demonstrate its merits for dealing with occlusions, viewpoints etc over 2D approaches. Then 2D approach [31] outperformed [82] by starting with well established 2D reasoning and aligning CAD models with 2D detections. The most recent work [83] outperformed [31] by a significant margin by introducing a 3D ConvNet model to encode 3D geometric features directly. So far, 3D centric sliding shapes leads the 3D detection performance on the challenging NYUV2 RGB-Depth dataset [79].

Although utilizing 3D geometric features for detection is promising, in practice the reconstructed 3D shapes are often incomplete (when projecting pixels of one single depth map back to 3D space), noisy and sparse (due to occlusions, reflections and absorptions of infrared lights). Hence, the quality of obtained surfaces is very different from that of CAD models with 360° panoramas, which makes fitting 3D bounding boxes to 3D points a very challenging task. In particular, when the majority of an object area on the depth map is in a ”black hole”, the recovered 3D shape hardly delivers salient features. However, light signals recorded in the 2D image plane are dense and structured, and humans still can perceive the objects and estimate their 3D locations from such images. Therefore, it should be possible to mimic the human 3D perception and leverage 2D image features directly using current deep learning techniques. As the proposed approach demonstrates this is indeed the case.

In this paper, we revisit the 3D amodal object detection problem from the 2D point of view. We start with 2D bounding box proposals obtained from extended multiscale combinatorial grouping (MCG) class independent object proposals [2, 35]. We design a novel 3D detection neural network based on Fast-RCNN framework that naturally integrates depth information with the corresponding visual appearances to identify object classes, orientations and their full extents simultaneously in indoor scenes, where 2D bounding boxes around superpixels together with RGB-Depth images are taken as inputs. To sum up, the highlights of the main contributions of this work are as follows:

- To the best of our knowledge, we are the first to reformulate the 3D amodal detection problem as regressing class-wise 3D bounding box models based on 2D image appearance features only.

- Given color, depth images and 2D segmentation proposals, we designed a novel 3D detection neural network that predicts 3D object locations, dimensions, and orientations simultaneously without extra step of training SVMs on deep features or fitting 3D CAD models to 2D detections.

- We do not make any Manhattan world assumption like 3D detectors do [82, 83] for orientation estimation, since objects in rooms are often cluttered and in disorder, reflecting various lifestyles and such assumptions may have dangerous consequences for autonomous systems like mobile robots.

- In addition, in order to benefit the future amodal 3D detection research, we improved the 3D ground-truth bounding boxes for the NYUV2 dataset by fixing many errors such as wrong labeling, partial extents, false negatives etc.

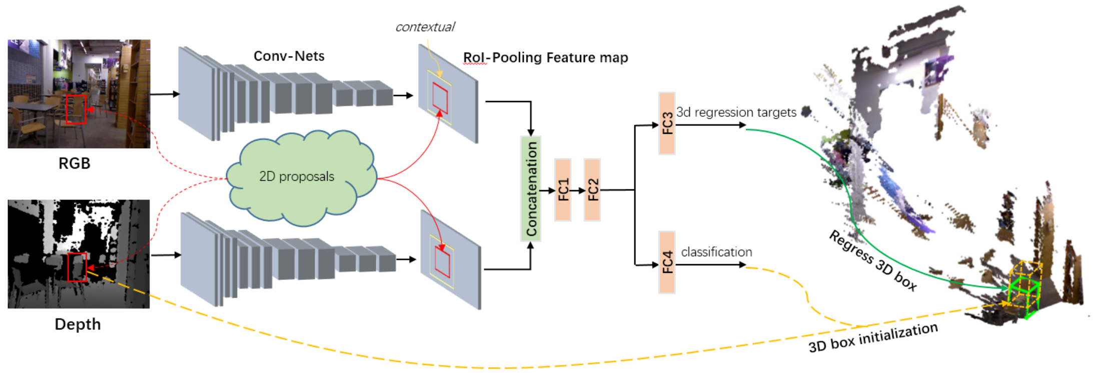


Figure 4.1: Overview of the proposed 3D object detection system. For each 2D segment proposal, we firstly initialize the localization of a 3D box (yellow dash box) based on depth information and its size according to classwise prior knowledge. Then object class and 3D regression offsets are jointly learned based on 2D features only, with the goal of obtaining the final 3D detection (green solid box) by adjusting the location, dimension, and orientation of the initial 3D box.

4.2 Related Works

Object detection is one of the oldest and most fundamental problems in computer vision. A huge number of works [91, 13, 17, 26] were proposed in the past few decades to infer bounding boxes around visible object parts within image planes. As human beings can effortlessly infer objects as a whole and complete, [48] took one step further towards obtaining similar levels of perception ability by addressing the full extent object inference problem on 2D image planes. Although this kind of object representation is richer than traditional modal inferences, it is still far from human perception level in the physical world and from the requirements for some robotic applications where robots are expected to interact with the environments. In order to fill the gap, an increased number of 3D object detection related research has been proposed, especially after active sensors become available in the consumer market. In the following, we briefly review the 3D detection algorithms for RGB-D images.

2D approaches in RGB-D images

2D approaches generally refer to methods where depth images are treated in a similar fashion as color images in traditional 2D detection task. [52] adapted the DPM algorithm to RGB-D images by utilizing the 3D euclidean distances from depth map. Handcrafted features were extracted from both color images within the output bounding boxes of existing 2D detectors and projected 3D point clouds within their associated foreground object segmentation masks. Their object locations are parametrized using 3D ellipsoids. [62] firstly generated 3D candidate cuboids by adapting the CPMC algorithm, and then incorporated 2D appearance features, object-object and object-scene context relationships into a Conditional Random Field (CRF) model for semantic labels inference.

Recently, feature engineering has been gradually replaced by deep Convolutional Neural Networks (CNNs) in 2D image based object detection. The most popular representative works are R-CNN [26], Fast-RCNN [24] and Faster-RCNN [25]. Inspired by [82], [63] adopt an exhaustive 3D sliding window strategy in the 3D point cloud where cross-modality features were extracted by feeding projected 2d bounding boxes to pretrained R-CNNs and the following bimodal deep Boltzman Machine trained separately. Detections were then determined by an ensemble of trained exemplar SVMs. Different from the previous sliding window framework, [35] built their detectors on the top of precomputed object segmentation candidates. They extended the R-CNN [26] to utilize depth information with HHA encoding (Horizontal Disparity, Height above ground and Angle of local surface normal w.r.t gravity direction). However, the outputs of their system were still limited to 2D bounding boxes. [31] extended [35] by firstly estimating 3D coarse poses for each detected 2D object and then aligning 3D CAD models to 3D points projected back from depth image that belongs to segmentation mask with Iterative Closest Point (ICP) algorithm.

The difference of the proposed method from the previous works above in three-folds: 1) no extra training examples or 3D CAD models are leveraged. 2) the model is trained end-to-end instead of piecewise. 3) no need for fitting point clouds lifted from depth map, which is often noisy and incomplete due to occlusions.

3D approaches in RGB-D images

3D approaches make use of depth map in a different way in that 3D points are reconstructed first, and the main processing is based on analyzing point clouds. [82] extended the traditional 2D sliding window strategy to 3D by putting 3D boxes within an estimated room box. A bunch of exemplar SVMs were trained with synthetic depths rendered from 3D CAD models, and then applied to each 3D detection window in a 3D indoor scene. 3D handcrafted features were built directly on discretized 3D space. Although the approach showed encouraging detection performance, the required computations are extremely expensive. [83] improved [82] dramatically with respect to both performance and efficiency by proposing 3D region candidates and extracting 3D features directly from 3D convolutional neural networks. Similar to [82], [72] designed handcrafted 3D features on point clouds for both 3D cuboid detection and Manhattan room layout prediction. In favor of better 3D features analysis, both [83] and [72] utilized enhanced depth map derived by fusing multiple nearby depth map frames to denoise and fill in missing depth. In contrast, our method naturally models the relationships between 2D features and 3D object localizations and full-extents in single frame RGB-D data.

4.3 3D Object Detection in RGB-D Images

4.3.1 Amodal 3D Object Detection

Given a pair of color and depth images, the goal of the amodal 3D object detection is to identify the object instance locations and its full extent in 3D space. As is well-known typical indoor environments in real life are very complicated, because objects may be heavily occluded and appear in a wide range of configurations. Encouraged by the success of 3D CAD model retrieval, the available depth map makes encoding 3D geometry features directly for detection very promising. However, the quality of depth map is far from perfect in reality due to measurement errors, and more importantly, the geometry of object instances is incomplete and its variations are determined by the camera view, e.g., see examples shown in Fig. 4.4. This may seriously limit the representation power from direct 3D reconstruction. Therefore, in this section we revisit the task of RGB-D amodal object detection and stick to the 2D representation by making the assumption that underlying relationships between 2D feature representations and 3D object locations and orientations exist. In the following, we explore how to effectively utilize RGB and depth for this task.

2D RoI proposals: Information contained in color and depth images are demonstrated to be complimentary to each other by varieties of RGB-D research works. While color encodes distinctive visual appearance features, depth conveys the geometric structures of objects. However, in 3D detection, one additional dimension significantly enlarges the search space. Since starting with well established 2D reasoning is arguably more efficient and accurate than starting from 3D reasoning [31]. We obtain the ROI proposals by using the adapted MCG algorithm in RGB-D images [35].

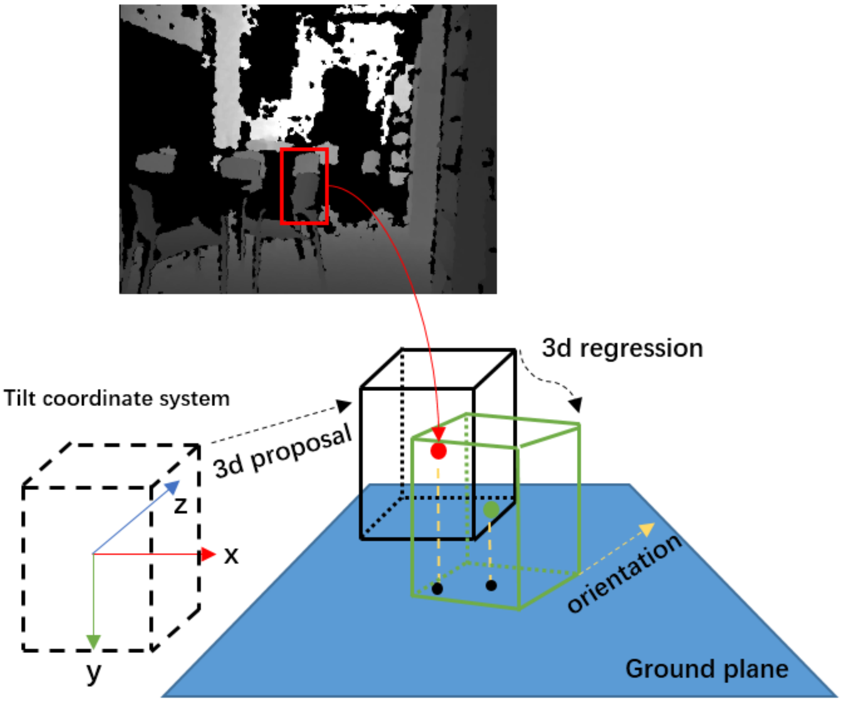


Figure 4.2: An example for the process of 3D box proposal and regression. The 3D box in dash line represents box initialized with class-wise averaged dimension in tilt coordinate system. The black solid line 3D box is translated from the dash-line box based on 2D segment. Finally, we regress the 3D box based on the features of the 2D segment to obtain the green 3D box. The yellow vector determines the orientation angle of 3D box to the principal axis (z-axis) in $\theta \in [-\pi/2, \pi/2]$, e.g., $\theta = 0$ if the yellow vector aligns with z-axis.

3D box proposal and regression:

Lifting 2D inferred object proposals to 3D bounding boxes by fitting a tight box around the 3D points projected from pixels in the instance segmentation [62, 31, 83] is not robust for 3D object detection due to both imperfect segmentations and noisy depth data. On the other hand, significantly extended 3D search space makes it inefficient to explore solutions in a brutal-force way [82]. One of the main contributions of this paper is initializing 3D proposals from 2D segments and reformulating the 3D amodal detection problem as regressing class-wise 3D bounding box models based on 2D visual appearance features only. As is shown in Figure 4.2, for each 2D segment proposal, we compute one 3D box counterpart as the 3D proposal. Then 3D proposals are transforming towards 3D ground truth according to learned high level 2D representations.

In this paper, the 3D bounding box is parametrized into one seven-entry vector $[x_{cam}, y_{cam}, z_{cam}, l, w, h, \theta]$. $[x_{cam}, y_{cam}, z_{cam}]$ is its centroid under camera coordinate system. $[l, w, h]$ represents its 3D size, and θ is the angle between principal axis and its orientation vector under tilt coordinate system (see Figure 4.2). The tilt coordinate system is converted from original camera coordinate system by aligning point clouds with gravity direction without any rotation around the y-axis:

$$XYZ_{cam} = R_{tilt}^{-1} * XYZ_{tilt} \tag{4.1}$$

$$R_{tilt} = R_x * R_z, \tag{4.2}$$

where R_{tilt} is the transform matrix between tilt and camera system, and R_x and R_z are rotation matrices around x-axis and z-axis, respectively.

3D box proposals are derived from corresponding 2D segment proposals. For box size in 3D proposals, we simply use averaged class-wise box dimensions estimated from training set as base 3D box size. It is better than fitting 3D points projected back from 2D segment pixels, which would significantly increase variance of box dimensions for regression. It is inspired by the cues of *familiar size* in human 3D perception [21, 48]. For example, when people are looking for an object like a bed, they have a rough object dimensions in their mind, which constraints the detection of bed instances. The center of proposed 3D box $[x_{ini}, y_{ini}, z_{ini}]$ is initialized based on 3D points projected from 2D segment pixels. Since depth maps are usually noisy and have missing data, we set z_{ini} to z_{med} which is the median depth value of segment points for the sake of robustness. In the case that the whole segment is a "black hole", we use interpolated depth map instead. x_{ini}

and y_{ini} are computed as described in Eq. (4.3): f is focal length of RGB camera, (o_x, o_y) is the principal point, (c_x, c_y) is the center of 2D box proposal.

$$\begin{cases} x_{ini} = z_{med} * (c_x - o_x)/f \\ y_{ini} = z_{med} * (c_y - o_y)/f \end{cases} \tag{4.3}$$

In contrast to [83], we do not make any Manhattan world assumption, since objects in rooms may appear in diverse orientations. In this work, the orientation angle θ is explicitly introduced as a parameter of 3D bounding box model. We define the orientation vector of a 3D box as the vector perpendicular to its longer edge in xz-plane (the yellow vector in Fig. 4.2). The initial orientation angle θ_{ini} is set to zero for all 3D box proposals, i.e., parallel to the x-axis in the tilt coordinate system, which is the case when box orientation vector aligns with camera principal axis. The range of θ is $[-\pi/2, \pi/2]$.

The 3D box regressor net will reshape the proposed raw 3D shape model based on the learned 2D appearance features. We represent the regression offsets as a 7-element vector $[\delta_x, \delta_y, \delta_z, \delta_l, \delta_w, \delta_h, \delta_\theta]$ for each positive example and ground truth boxes during training stage. Instead of finding the closest matching of major directions between detected box and ground-truth boxes [83] for computing box dimension differences, we can directly compare corresponding length, width and height parameters and normalize them by the size of the detected box, which is possible due to our parameterization of 3D bounding boxes. Similar to [24], the target for learning is then normalized by statistical information from proposed boxes.

Multi-task Loss: Each training example is associated with a ground-truth class c and corresponding ground-truth 3D bounding box. To jointly train for classification and bounding box regression, the loss function is defined as follows:

$$L(p, c, t_{3d}^{c}, v_{3d}) = L_{cls}(p, c) + \lambda(c >= 1)L_{3d}(t_{3d}^{c}, v_{3d}), \tag{4.4}$$

where t_{3d}^{c} expresses the regression offsets w.r.t ground truth locations, v_{3d} are regression targets, p is the predicted probability of the object class, L_{cls} is defined as softmax function, and L_{3d} is $L1$ smooth loss as defined in [24].

Post processing: We apply typical Non-Maximum Suppression (NMS) scheme to the 2D detected boxes. No NMS is used in 3D. In contrast to [83], we do not perform any further pruning of the results, e.g., based on object size statistics.

4.3.2 Convolutional Network Architecture

There have been many deep convolutional network models proposed recently for 2D image based recognition. In this paper, we adopt the Fast-RCNN [24] as the raw base model due to both of its one single stage training architecture and high efficiency by sharing features computation. As is shown in Figure 4.1, color and depth images go through two VGG16 [80] Conv-Nets for computing shared feature maps, respectively. Features extracted from RoI pooling layer based on 2D object proposals and their enlarged contextual patches are concatenated for multiple tasks learning.

Mini-batch sampling

For training deep neural network models, a small set of examples is randomly selected from training set to update model parameters at each iteration for the sake of computation efficiency. It is very important to properly define and select positive and negative examples from RoI pool for image based object detections.

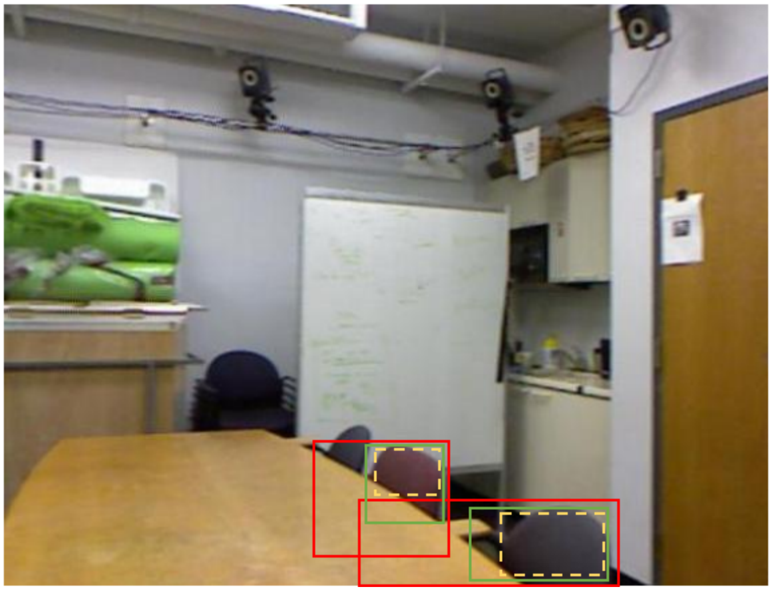


Figure 4.3: Red: two examples of 2D ground truth bounding boxes from [81]. Green: 2D RoI proposals. If compared 2D RoI proposals directly to red bounding boxes, the two positive chair examples are wrongly treated as negative ones. To solve this problem, we added yellow (dashed) $gt2d_{sel}$ boxes for mini-batch sampling.

Typically, one RoI is treated as positive if it has intersection over union (IoU) overlap with ground truth box greater than 0.5, and negative if IoU is between 0.1 and 0.5. However, directly applying this rule to mini-batch sampling using 2D annotations provided by [81] would cause a serious problem. [81] provides two kinds of 2D ground truth bounding boxes for NYUV2 dataset: 1) projected 2D bounding boxes by fitting visible point clouds, and 2) projected 2D bounding boxes from amodal 3D bounding boxes. Using either kind for mini-batch sampling with 2D representations, the detection performance degrades dramatically since the true positive segments may be treated as negative ones if comparing them directly to the 2D ground truth provided by [81], as is illustrated in Fig. 4.3.

To fix the problem, we added new 2D ground truth box named $gt2d_{sel}$ to the training set for determining positive and negative examples from proposed 2d segments only. We stress that the amodal 2D bounding boxes provided by [81] can be still used as targets for the 2D box regression task.

Each mini-batch consists of 256 randomly selected 2D box proposals from $N = 2$ images (128 RoIs per image). The ratio of positive and negative examples is set to $1 : 3$.

For data augmentation, we flip horizontally images and their corresponding 3D bounding boxes. No other extra data is used during training.

4.4 Improved 3D annotations on NYUV2

The labeled NYU Depth V2 dataset [79] is a most popular but very challenging dataset in the RGBD scene understanding research community. The original version provides 1449 RGB-Depth indoor scene images with dense 2D pixelwise class labels. To enrich the labeling features and encourage 3D object detection research, in the SUN RGBD dataset [81] (superset of NYUV2) Xiao et al. added extra 3D bounding boxes and room layouts to ground truth annotations. Since depth maps are imperfect in reality due to measurement noise, light reflection and absorption, and occlusion etc, they also refined the quality of depth maps by integrating multiple RGB-D frames from the NYUV2 raw video data.

However, the extended 3D annotations in [81] have some notable issues: 1) 3D boxes were labeled independently from the original 2D object instances in NYUV2. This inconsistency leads to many salient objects being unlabeled or mislabeled, which causes unnecessary false negatives during the detection task. 2) 3D amodal annotations are mixed with modal ones. Amodal bounding boxes cover the full-extent of objects, while modal ones only encompass the visible parts (e.g., see the beds in Figure 4.4). This is a very undesirable feature for the "amodal" detection as perused in this paper following the approaches in [5, 48, 83]. 3) Inconsistent labelings among scenes that have overlapping areas. 4) Inaccurate 3D extents or locations of object instances.

In order to provide better 3D labelings for amodal 3D object detection re-

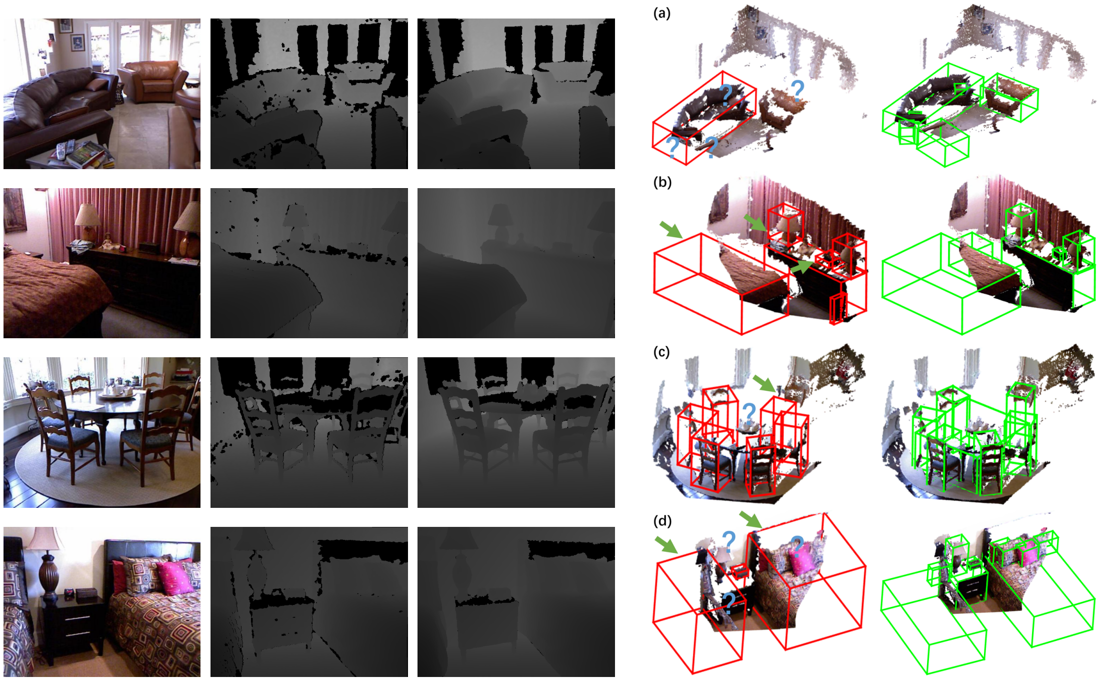


Figure 4.4: Examples of improved 3D annotations for 19 amodal 3D object detection classes and comparisons with annotations in SUN RGBD dataset [81]. Column 1: color images. Column 2: original single frame raw depth maps. Column 3: refined depth maps by integrating multiple depth maps within nearby video frames. Column 4: Ground truth 3D bounding boxes (red color) from [81]. Blue question marks represent missing object annotations. Green arrows point to problematic object annotations. Column 5: our improved 3D annotations (green color). As is shown in Column 4, notable issues include missing bounding boxes for salient objects, e.g., 2 sofas, 1 table and 1 pillow are missing in (a), 1 table is missing in (c), 1 lamp, 1 nightstand and several pillows are missing in (d), modal boxes for partial visible objects are incomplete, e.g., all bounding boxes for beds in (b) and (d), inaccurate 3D extensions and locations, e.g., 1 chair in (c) is mis-located, 1 lamp in (b) is floating above table surface, 1 box object in (b) has very loose bounding box. In comparison to examples shown in Column 4, we provide much more reasonable annotations for amodal 3D object detection research purpose. In this paper, we use original single frame depth maps as in column 2 as input instead of refined ones that were adopted in [83].

Methods	🛁	🛏	📚	📦	🪑			🚪		🗑
[83](old gt3d)	64.4	82.3	20.7	4.3	60.6	12.2	29.4	0.0	38.1	27.0
[83]	62.3	81.2	23.9	3.8	58.2	24.5	36.1	0.0	31.6	27.2
Ours	36.1	84.5	40.6	4.9	46.4	44.8	33.1	10.2	44.9	33.3

Methods	🪔	🖥			🚰	🛋		📺	🚽	**mAP**
[83](old gt3d)	22.1	0.7	49.5	21.5	57.8	60.5	49.8	8.4	76.6	36.1
[83]	28.7	2.0	54.5	38.5	40.5	55.2	43.7	1.0	76.3	36.3
Ours	29.4	3.6	60.6	46.3	58.3	61.8	43.2	16.3	79.7	**40.9**

Table 4.1: 3D Object Detection Performance Comparisons on 19 Classes on NYUV2 dataset. 1st row is evaluated using 3D annotations in [81]. The others are evaluated using the improved 3D annotations (see Sec 4.4).

search, we provide improved ground truth 3D bounding boxes annotations for 19 indoor object classes (bathtub, bed, bookshelf, box, chair, counter, desk, door, dresser, garbage bin, lamp, monitor, nightstand, pillow, sink, sofa, table, tv, toilet) by complying with the following stricter principles: 1) Amodal for all: all the 3D bounding boxes should encompass the whole 3D instance of the object, even if only object parts are visible. 2) Place tight boxes around 3D object extents with reasonable orientations. 3) Comply with the physical configuration rules. For example, table and chair rest on the floor, and the height of door should not be too short. 4) Labeling is as consistent as possible with the NYUV2 2D object instances.

In the improved annotation set, we provide 3D amodal bounding boxes, 2D amodal bounding boxed cropped by image plane and rotation matrix R_{tilt} for gravity alignment etc. Some examples and comparisons with annotations in [81] are shown in Figure 4.4. The improved annotations will be released on the authors' website.

Methods	🛁	🛏	📚	📦	🪑			🚪		🗑
img	27.9	64.5	24.5	1.5	33.1	46.0	20.3	1.7	28.7	32.1
img+HHA	33.1	83.9	29.8	6.0	43.1	46.3	25.3	1.87	30.9	32.9
img+d	38.9	85.2	37.5	11.4	46.5	47.1	29.9	4.2	43.3	37.3
img+d+ct	36.1	84.5	40.6	4.9	46.4	44.8	33.1	10.2	44.9	33.3
img+d+ct-3dreg	8.3	5.0	14.3	2.1	14.1	3.6	0.6	0.7	4.1	29.5

Methods	💡	🖥			🚰	🛋		📺	🚽	**mAP**
img	24.6	3.0	43.4	27.7	49.6	46.7	27.6	1.3	66.0	30.0
img+HHA	24.3	4.1	58.3	40.3	54.8	59.6	39.6	3.5	69.5	36.2
img+d	30.8	1.3	59.8	44.1	57.7	63.8	39.4	11.6	75.5	40.1
img+d+ct	29.4	3.6	60.6	46.3	58.3	61.8	43.2	16.3	79.7	40.9
img+d+ct-3dreg	27.1	2.4	23.0	31.4	20.5	34.5	4.6	1.7	67.6	15.5

Table 4.2: Ablation study on NYUV2 dataset. ”img”: use color image only as input to our detection network. ”HHA”: depth embedding of [35]. ”d”: normalized depth map. ”ct”: context information. ”3dreg”: 3d regression offsets. ”+”: with. ”-”: without.

4.5 Experiments

In this section, we compare our algorithm with the currently best performing 3D detector [83] on the NYUV2 dataset [79] with the improved 3D bounding box annotations as described in Sec 4.4. Control experiment analysis and related discussions are also provided for better understanding the importance of each component in the designed 3D detection system. In the standard NYUV2 dataset split, the training set consists of 795 images and test set contains 654 images. We follow this standard for all the experiments. For our algorithm, we use the single frame depth map provided by the NYUV2 instead of the refined version in SUN-RGBD dataset.

3D Amodal Detection Evaluation

In order to compare the proposed approach to *deep sliding shapes* [83], we perform evaluation on 19 object classes detection task. We evaluate the 3D detection performance by using the 3D volume intersection over union (IoU) metric firstly defined in [82]. A detected bounding box is considered as a true positive if the IoU score is greater than 0.25. In the experiment, we set λ to 1 in the loss function. We use momentum 0.9, weight decay 0.0005, and "step" learning rate policy in Caffe, where base learning rate is 0.005, and γ is 0.1. We run SGD for 40000 mini-batch iterations during the training stage. In order to reduce the internal covariate shift, we normalized activations by adding BatchNorm Layers [44] to the 3D detection network.

In Table 4.1, we quantitatively compare with the state-of-the-art 3D approach algorithm "deep sliding shape" [83] on a 19-class detection task on the NYUV2 RGB-D dataset. Our method significantly outperforms [83] by a clear margin **4.6%** measured by mean Average Precision score (mAP). In particular, we achieve much better detection performances on difficult object categories reported in [83] such as door, tv, box, monitor. The reason is that in [83] the 3D box proposals network (RPN) relies on the quality of recovered 3D point cloud. But, in practice, the depth data from Kinect alike sensors are noisy and incomplete. Therefore, if the point cloud is sparse or empty for object instances such as tv or monitor, then the corresponding 3D anchor boxes are treated as negative 3D proposals and discarded. In contrast, our approach is more robust in such cases, since our 3D box initialization uses median value of segment pixel depths and 3D regression are based on learned 2D features (see Sec. 4.3.1), and hence neither depend on density nor geometries of 3D point clouds.

In addition, we list the results of [83] evaluated on the 3D annotations of [81] as a reference. Their results based on the improved 3D annotations are slightly better, which might be due to the fact that wrong labelings have been corrected in the new annotations.

We also provide qualitative results in Figure 4.5 and 4.6. True positive detections in Figure 4.5 indicate that 2D representation features are useful for detecting 3D objects with various of orientations, sizes and locations in complexed indoor scenes. In Figure 4.6, we list several failure cases including wrong box size, inaccurate locations, wrong box orientations, and mis-classifications.

Ablation Study

To understand the importance of each component of our system, we conduct control experiments and list detection results in Table 4.2. We are reaching the following conclusions: 1) Color images contain rich 2D features for inferring object 3D full-extents. 2) The features encoded in depth map are complimentary to those in color images. 3) We normalized the depth map by truncating depth value beyond 8 meters. It achieves 2.9% improvement than using HHA embedding as Horizontal disparity, Height above ground and Angle of local surface normal with inferred gravity direction. 4) Contextual information slightly improves the performance by 0.8%.

In order to demonstrate effectiveness of 3D regression learned by the proposed system, we remove 3D offsets and evaluate the initial 3D boxes in ”img+d+ct-3dreg”. The performance degrades dramatically by 25.4%.

Computation Speed

Our 3D detection system is developed based on the open source Caffe CNN library [47]. The training of 3D detector takes around 15 hours on an Nvidia Titan X GPU using CUDA 7.5 and cuDNN v4 support. The GPU usage is around 9 GB. During testing, the detection net takes 0.739s per RGB-D image pair, which is nearly **20x** faster than the Object Recognition Network (ORN) in [83].

4.6 Conclusion

We present a novel amodal 3D object detection system that directly learns deep features in RGB-D images without performing any 3D point reconstruction. Hence our system learns 2D visual appearance features from pairs of color and depth images. Experiments demonstrate that the 2D visual features are correlated to 3D object sizes, locations, and orientations. Our approach significantly outperforms the best performing 3D detector [83], which is truly a 3D approach, since it analyzes 3D point clouds.

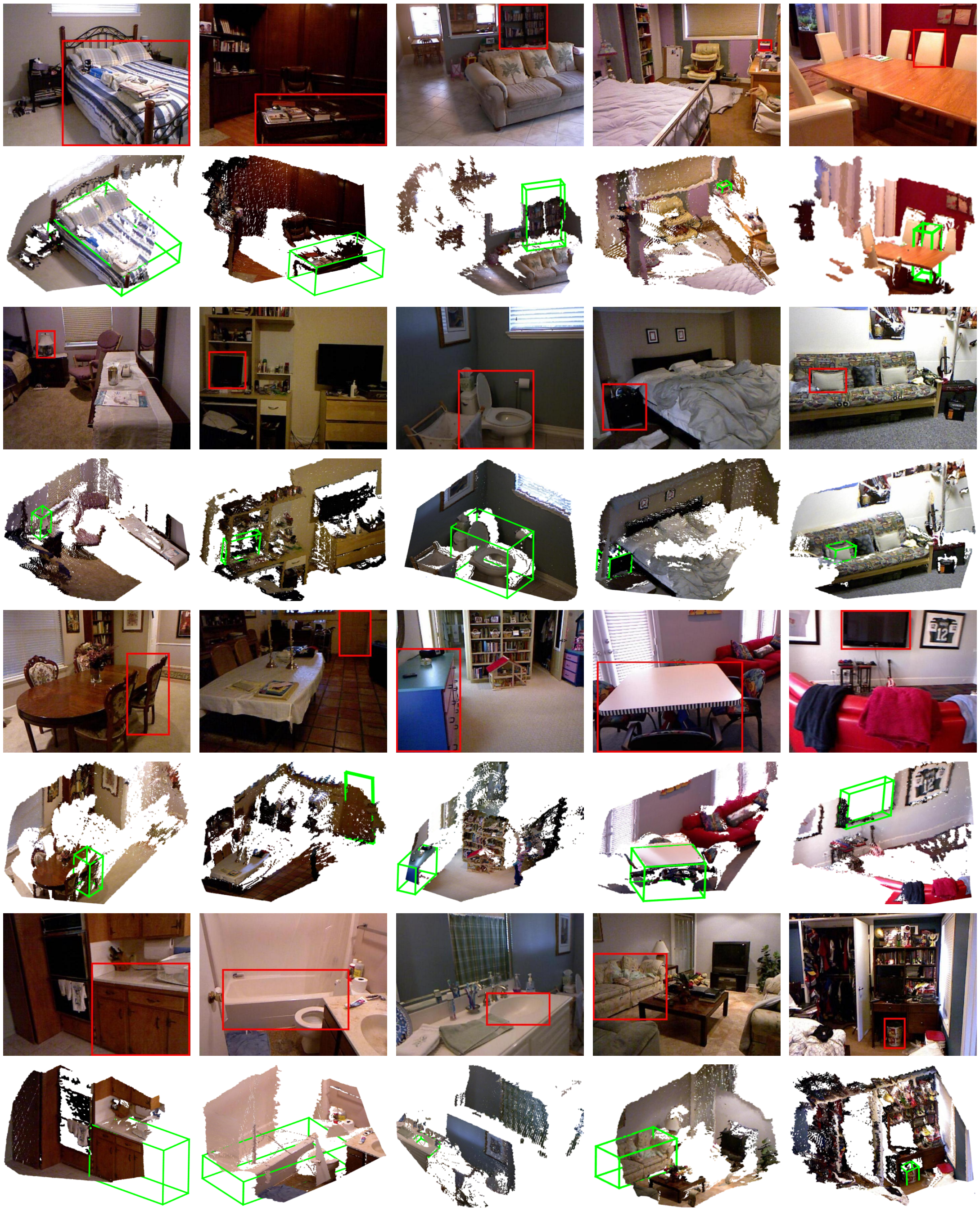


Figure 4.5: Examples of detected true positive 3D amodal bounding boxes on NYUV2. 3D detections are rendered in 3D space in green. The corresponding object 2D locations are marked with red bounding boxes.

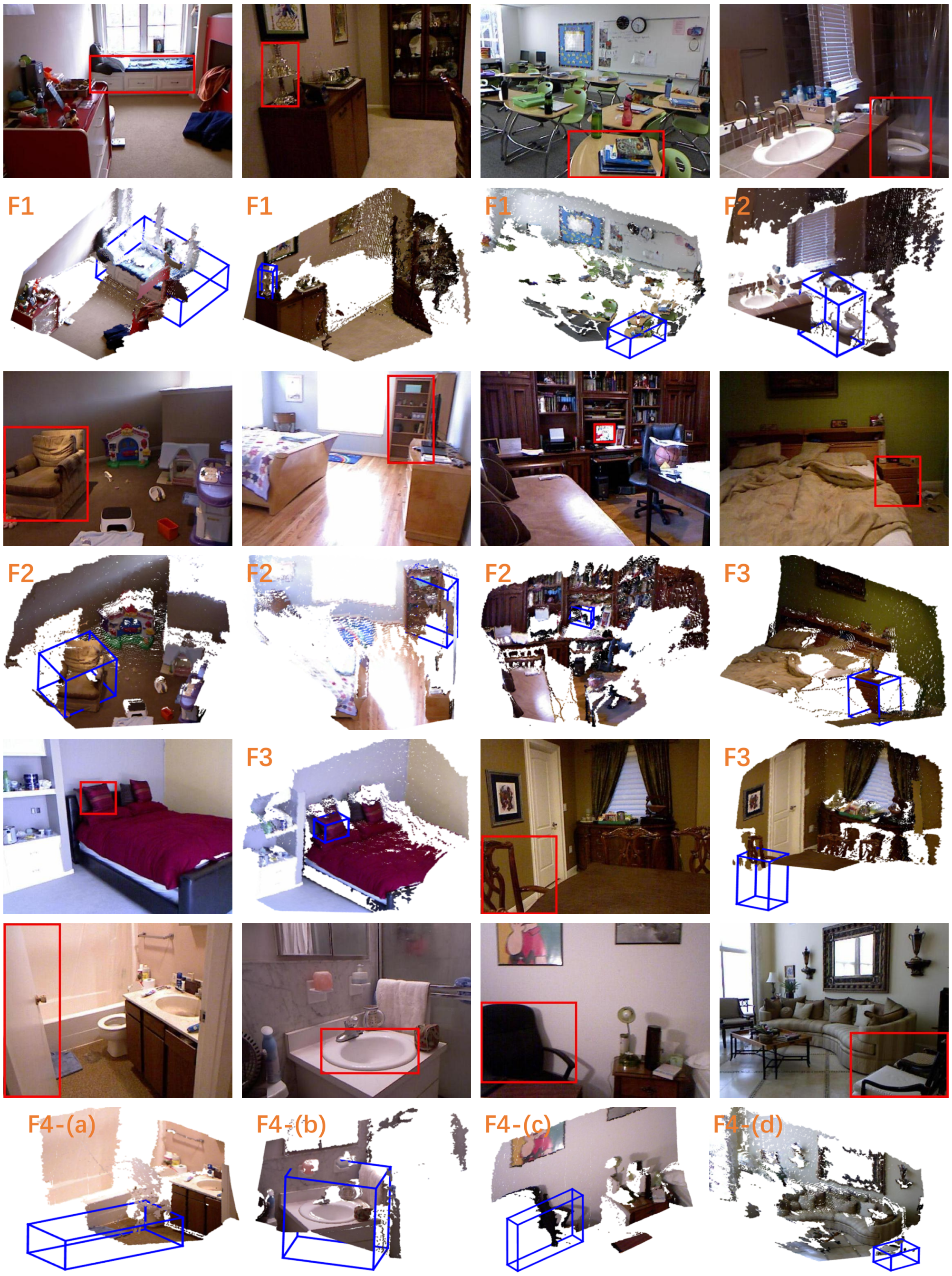


Figure 4.6: Examples of failure cases. 3D detections are rendered in 3D space in blue. The corresponding objects are marked with red bounding boxes. We show four types of failures. F1: box dimension errors. F2: orientation errors. F3: 3D location errors. F4: classification errors ((a) door detected as bathtub, (b) sink detected as toilet, (c) chair detected as tv, (d) chair detected as table).

BIBLIOGRAPHY

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011.

[2] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014.

[3] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *Advances in neural information processing systems*, pages 244–252, 2010.

[4] Y. Boykov and G. Funka-Lea. Graph cuts and efficient nd image segmentation. *International journal of computer vision*, 70(2):109–131, 2006.

[5] T. P. Breckon and R. B. Fisher. Amodal volume completion: 3d visual completion. *Computer Vision and Image Understanding*, 99(3):499–526, 2005.

[6] W. Brendel and S. Todorovic. Segmentation as maximum-weight independent set. In *Advances in Neural Information Processing Systems*, pages 307–315, 2010.

[7] C. R. Brice and C. L. Fennema. Scene analysis using regions. *Artificial intelligence*, 1(3):205–226, 1970.

[8] J. Carreira and C. Sminchisescu. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1312–1328, 2012.

[9] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pages 2147–2156, 2016.

[10] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

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

[12] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 1124–1131. IEEE, 2005.

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[14] Y. Deng, B. S. Manjunath, and H. Shin. Color image segmentation. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[15] J. Dong, Q. Chen, S. Yan, and A. Yuille. Towards unified object detection and semantic segmentation. In *European Conference on Computer Vision*, pages 299–314. Springer, 2014.

[16] C. Erdogan, M. Paluri, and F. Dellaert. Planar segmentation of rgbd images using fast linear fitting and markov chain monte carlo. In *Computer and Robot Vision (CRV), 2012 Ninth Conference on*, pages 32–39. IEEE, 2012.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[18] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[19] S. Fidler, S. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *Advances in neural information processing systems*, pages 611–619, 2012.

[20] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[21] J. Fredebon. The role of instructions and familiar size in absolute judgments of size and distance. *Perception & Psychophysics*, 51(4):344–354, 1992.

[22] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí. Yet another survey on image segmentation: Region and boundary information integration. In *European Conference on Computer Vision*, pages 408–422. Springer, 2002.

[23] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[24] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[25] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[26] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[27] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. Segmentation driven object detection with fisher vectors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2968–2975, 2013.

[28] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *International Journal of Computer Vision*, 80(3):300–316, 2008.

[29] L. Guan, T. Yu, P. Tu, and S.-N. Lim. Simultaneous image segmentation and 3d plane fitting for rgb-d sensorsan iterative framework. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 49–56. IEEE, 2012.

[30] A. Gupta, M. Hebert, T. Kanade, and D. M. Blei. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Advances in neural information processing systems*, pages 1288–1296, 2010.

[31] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015.

[32] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision*, 112(2):133–149, 2015.

[33] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 564–571, 2013.

[34] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.

[35] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.

[36] D. Hähnel, W. Burgard, and S. Thrun. Learning compact 3d models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15–27, 2003.

[37] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312. Springer, 2014.

[38] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *2009 IEEE 12th international conference on computer vision*, pages 1849–1856. IEEE, 2009.

[39] V. Hedau, D. Hoiem, and D. Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *European Conference on Computer Vision*, pages 224–237. Springer, 2010.

[40] V. Hedau, D. Hoiem, and D. Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *European Conference on Computer Vision*, pages 224–237. Springer, 2010.

[41] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 654–661. IEEE, 2005.

[42] D. Hoiem, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from an image. *International Journal of Computer Vision*, 91(3):328–346, 2011.

[43] S. L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *Journal of the ACM (JACM)*, 23(2):368–388, 1976.

[44] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[45] A. Ion, J. Carreira, and C. Sminchisescu. Image segmentation by figure-ground composition into maximal cliques. In *2011 International Conference on Computer Vision*, pages 2110–2117. IEEE, 2011.

[46] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3d object dataset: Putting the kinect to work. In *Consumer Depth Cameras for Computer Vision*, pages 141–165. Springer, 2013.

[47] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[48] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Amodal completion and size constancy in natural scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 127–135, 2015.

[49] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri. Geometry driven semantic labeling of indoor scenes. In *European Conference on Computer Vision*, pages 679–694. Springer, 2014.

[50] K. Khoshelham. Accuracy analysis of kinect depth data. In *ISPRS workshop laser scanning*, volume 38, page W12, 2011.

[51] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012.

[52] B.-s. Kim, S. Xu, and S. Savarese. Accurate localization of 3d objects from rgb-d data using segmentation hypotheses. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pages 3182–3189, 2013.

[53] P. Kohli, P. H. Torr, et al. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82(3):302–324, 2009.

[54] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *Advances in Neural Information Processing Systems*, pages 244–252, 2011.

[55] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Graph cut based inference with co-occurrence statistics. In *European Conference on Computer Vision*, pages 239–253. Springer, 2010.

[56] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[57] C. Lang, T. V. Nguyen, H. Katti, K. Yadati, M. Kankanhalli, and S. Yan. Depth matters: Influence of depth cues on visual saliency. In *Computer Vision–ECCV 2012*, pages 101–115. Springer, 2012.

[58] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2136–2143. IEEE, 2009.

[59] V. Lempitsky, A. Vedaldi, and A. Zisserman. Pylon model for semantic segmentation. In *Advances in neural information processing systems*, pages 1485–1493, 2011.

[60] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 689–694. ACM, 2004.

[61] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d ob-

ject detection with rgbd cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1417–1424, 2013.

[62] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1417–1424, 2013.

[63] W. Liu, R. Ji, and S. Li. Towards 3d object detection with bimodal deep boltzmann machines over rgbd imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3013–3021, 2015.

[64] T. Ma and L. J. Latecki. Maximum weight cliques with mutex constraints for video object segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 670–677. IEEE, 2012.

[65] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 416–423. IEEE, 2001.

[66] M. Meilă. Comparing clusterings by the variation of information. In *Learning theory and kernel machines*, pages 173–187. Springer, 2003.

[67] F. Meyer. Color image segmentation. In *Image Processing and its Applications, 1992., International Conference on*, pages 303–306. IET, 1992.

[68] N. Payet and S. Todorovic. From contours to 3d object detection and pose estimation. In *2011 International Conference on Computer Vision*, pages 983–990. IEEE, 2011.

[69] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak. Fast plane detection and polygonalization in noisy 3d range images. In *2008 IEEE/RSJ Inter-*

national Conference on Intelligent Robots and Systems, pages 3378–3383. IEEE, 2008.

[70] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 413–420. IEEE, 2009.

[71] X. Ren, L. Bo, and D. Fox. Rgb-(d) scene labeling: Features and algorithms. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2759–2766. IEEE, 2012.

[72] Z. Ren and E. B. Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *Proceedings of the IEEE International Conference on Computer Vision*, 2016.

[73] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[74] A. Roy and S. Todorovic. Scene labeling using beam search under mutex constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1178–1185, 2014.

[75] R. B. Rusu. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, 24(4):345–348, 2010.

[76] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[77] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[78] N. Silberman and R. Fergus. Indoor scene segmentation using a structured

light sensor. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 601–608. IEEE, 2011.

[79] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012.

[80] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[81] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 567–576, 2015.

[82] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *European Conference on Computer Vision*, pages 634–651. Springer, 2014.

[83] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[84] J. Strom, A. Richardson, and E. Olson. Graph-based segmentation for colored 3d laser point clouds. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2131–2136. IEEE, 2010.

[85] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 945–953, 2015.

[86] C. J. Taylor and A. Cowley. Segmentation and analysis of rgb-d data. In *RSS 2011 workshop on RGB-D cameras*, volume 90, 2011.

[87] C. J. Taylor and A. Cowley. Parsing indoor scenes using rgb-d imagery. In *Robotics: Science and Systems*, volume 8, pages 401–408, 2013.

[88] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[89] R. Unnikrishnan, C. Pantofaru, and M. Hebert. A measure for objective evaluation of image segmentation algorithms. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*, pages 34–34. IEEE, 2005.

[90] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.

[91] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[92] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[93] J. Xiao, A. Owens, and A. Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1625–1632, 2013.