Unified Models for Recovering Semantics and Geometry from Scenes

by

Byung-soo Kim

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
2014

Doctoral Committee:

Assistant Professor Silvio Savarese, Co-Chair, Stanford University
Assistant Professor Honglak Lee, Co-Chair
Professor Anna C. Gilbert
Senior Researcher Pushmeet Kohli, Microsoft Research Cambridge
Associate Professor Gregory H. Wakefield

This doctoral dissertation is dedicated to my ever supportive parents, loving wife and daughters.

ACKNOWLEDGEMENTS

I would never have been able to finish my dissertation without the guidance of my committee members, help from friends, and support from my family.

Firstly and most importantly, I would like to express my deepest gratitude to my advisor Prof. Silvio Savarese. I was lucky to benefit from his extensive knowledge of computer vision and sincere passion toward research. I also appreciate his trust and support for the last 5 years when I grew from a clueless student to a computer vision researcher.

During my previous projects, I was lucky to collaborate with several distinguished researchers and professors. I am grateful for Dr. Pushmeet Kohli for sharing his deep knowledge of discrete optimization methods through many previous projects. It was a great pleasure to work with Prof. Anna C. Gilbert, who showed me how sparse approximation methods can be used for computer vision applications. I thank Prof. Honglak Lee for his feedback about the mathematical models in this thesis, and also for sharing tips for the possible future careers.

I would like to thank many people I met in Michigan; Min Sun for sharing his insight in research and life, Wongun Choi for discussing my random thoughts, Yingze Bao for his valuable feedbacks about RGB-D research projects, Yu Xiang for helpful discussions about graphical models and object detection schemes, Shili Xu for working hard on our CVPR'13 project, and Ryan Tokola for helpful feedbacks about research and presentations. It was a great pleasure to share the office with Liang Mei and Johnny Chao for my first and last two years. I cannot forget Becky Turanski for her

abiding kindness over the last 5 years.

I thank to my friends for making my days in Ann Arbor happy – Hyunjung Cho, Seunghyun Oh, Jae Young Park, Donghwan Kim, Kihyuk Sohn, Daeyeon Jung, Changkyu Song, and Jung Kuk Kim.

Finally, I truly appreciate the support from my family. I am grateful for my parents' unconditioned love ever since I was born. I thank my parents-in-law for believing in me enough to allow me to study abroad with their precious daughter. My sister and brother-in-law were real supports for my family when I was studying abroad. Needless to say, I thank to my wife and daughters for being the reason of my happiness – I am very lucky to see their smiles everyday.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Unified Models for Recovering Semantics and Geometry from Scenes

by

Byung-soo Kim

Chair: Prof. Silvio Savarese

Understanding contents of an image, or *scene understanding*, is an important yet very challenging problem in computer vision. In the last few years, substantially different approaches have been adopted for understanding "*things*" (object categories that have a well defined shape such as people and cars), "*stuff*" (object categories that have an amorphous spatial extent such as grass and sky), and the "*geometry*" of scenes.

In this thesis, we propose coherent models for the simultaneous recognition of *things*, *stuff*, and *geometry*. The key contributions are *i)* to model their individual properties as well as relative properties, and *ii)* to propose a coherent framework that efficiently solves complicated tasks for scene understanding. We demonstrate that each task can be improved by also solving the other tasks in a joint fashion. The proposed models are capable of handling different types of inputs such as RGB, RGB-D, or hierarchically organized images.

We have carried out extensive quantitative and qualitative experimental analysis to demonstrate the effectiveness of our theoretical findings and showed that our

approaches yield competitive performances with respect to state-of-the-art methods.

CHAPTER I

Introduction

In this chapter, we introduce the scene understanding problem and relevant challenges in Sec. 1.1. The previous efforts for scene understanding are summarized in Sec 1.2. We highlight our contributions toward solving the scene understanding problem in Sec. 1.3.

1.1 Scene Understanding Problem

The goal of scene understanding is to develop an automated recognition system that can recognize a scene as effectively as humans do. This problem has been studied by the early vision community [11, 77, 109] due to its potential use in a number of applications such as robotics, human computer interaction, and autonomous driving.

Automatically understanding a scene from images is an extremely challenging problem; as the adage "A picture is worth a thousand words" says, there are an exponential number of ways to interpret a scene based on the observer's interest and criteria. In order to make the problem tractable, researchers have identified a number of key recognition sub-tasks: *i)* object detection to localize an object, *ii)* classification to identify a semantic label of the region of interest, *iii)* semantic labeling to classify each pixel, and *iv)* geometry estimation to understand the physical structure of the scene. For example, while looking at Fig. 1.1, one is able to localize object (detection)

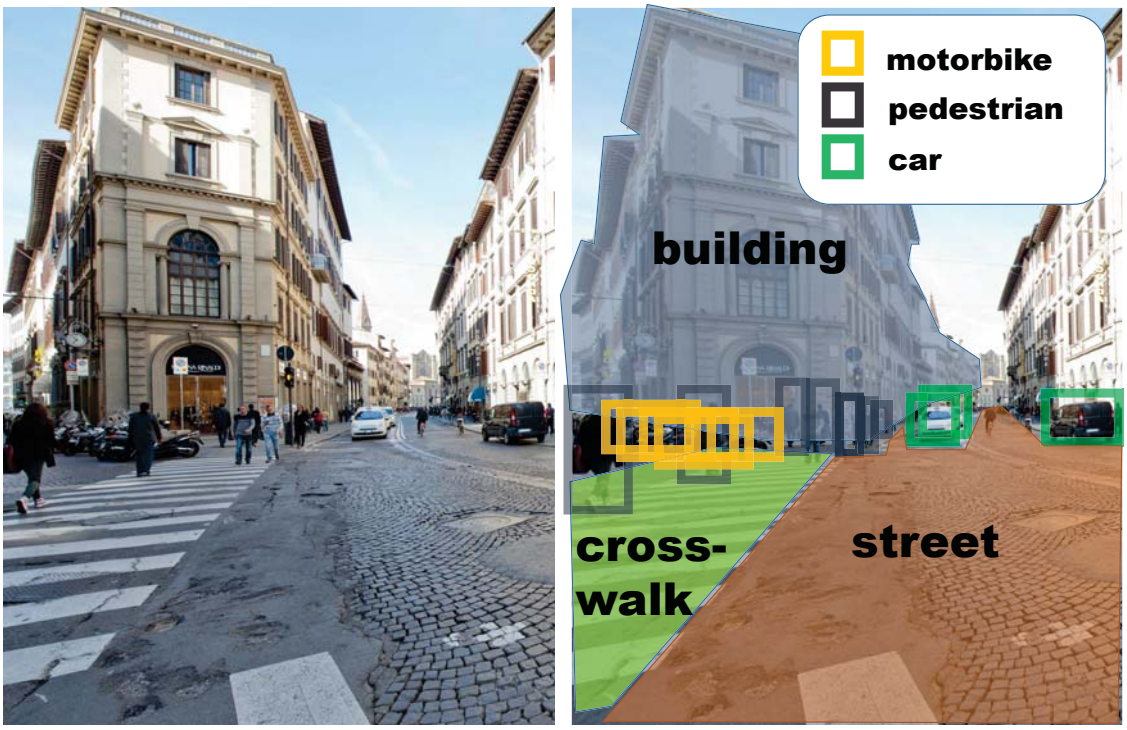


Figure 1.1: This figure highlights different sub-tasks of the scene understanding problem. Within a second, human can localize objects (detection) such as pedestrians or cars (classification), as well as to understand its background as a sidewalk or building (semantic labeling) and the geometric structure this scene is composed of (geometry estimation).

such as a pedestrian (classification), identify the background as sidewalk (semantic labeling), and determine the orientation of the building facade (geometry estimation).

A number of methods have been proposed for each of these tasks (Sec. 1.2), and there have been meaningful achievements in areas such as pedestrian detection and image classification. While most of these works solve these tasks in isolation, we believe that an integrated approach where detection, segmentation, and geometry estimation can be solved jointly can be highly beneficial. For example, in Fig. 1.1, by segmenting an image (labeling each pixel as building, crosswalk, or street), geometric properties (vertical facade of building or horizontal support of street and crosswalk) can be inferred. Similarly, by solving the detection problem (cars and pedestrian), we can estimate geometric properties (empty space or occupied) or solve segmentation

problem (street or crosswalk).

Modeling the correlation between different recognition tasks is of the main objective of this thesis, and we propose unified models to understand scenes by focusing on various subtasks with different types of sensor data. Specifically, we first propose a coherent model based on Conditional Random Fields (CRF) where object detection and semantic labeling problems are solved in a joint fashion. Our models are evaluated on the cluttered images, which are extremely challenging for models that consider detection and semantic labeling separately. Second, the CRF model is extended to represent 3D space so that it can jointly reconstruct a 3D scene and perform semantic labeling tasks, in the case of using RGB-D data. As key building blocks for such holistic models, we have also proposed methods that enable accurate object localization in 3D, and that exploit semantic hierarchies to recognize objects at various degrees of semantic granularity.

1.2 Previous Works

We categorize previous efforts for scene understanding based on based on whether they focus on semantic inference or geometry estimation.

1.2.1 Semantic understanding

The problem of semantic understanding aims at generating semantic labels for a region of interest. Although the domain of semantic labels can be extensive, we follow definition introduced by [39, 63, 81], and divide semantic labels into two categories: *things* and *stuff*. *Things* indicate object classes having rigid shape such as cars or pedestrians, whereas *stuff* indicates classes having more amorphous shapes such as streets or buildings. Different strategies are proposed for each things and stuff recognition.

Things recognition. An object that belongs to *thing* class often occupies a limited

spatial extent in an image space, such as a region in a bounding box. Therefore, one paradigm for things recognition is to search for a location in an image space where the observation and the pre-trained model of a specific object class closely match [36, 37, 41, 87, 127]. When more than one thing class exists in the scene, several methods have been proposed in order to balance confidence in the different classes [28, 105].

Compared to detection methods, classification methods are typically used in the case when a region of interest is given, for example, a bounding box or whole image space. A variety of different techniques have been proposed with sequential steps for classification; feature extraction [6, 24, 89, 119], pooling [86, 124, 128], modeling [14, 15, 19, 100], and training [32, 121, 129].

Stuff recognition. Since the spatial extent of stuff type classes is less rigid than thing type objects, the detectors for thing type objects cannot be used to recognize stuff. Instead, a number of methods are proposed to associate each image element (*e.g.,* pixel or super-pixel) with a specific object class label by looking at its appearance as well as the appearance of the surrounding area.

A majority of these methods ([58, 72, 76]) are built as conditional random field (CRF) models, where image elements are represented as vertices in a graph and adjacency among image elements is represented using edges.

1.2.2 Geometric understanding

For many applications such as robotics or autonomous navigating system, understanding the geometry within an unknown environment is required. Early works such as [29, 50] proposed methods to build and update a map and keep the track of camera location. However, these methods can only be used with sensors such as laser, Lidar, sonar sensor, or multiple 2D cameras.

Structure from motion (SfM) techniques based on multiple 2D single-view RGB

Figure 1.2: The relationship between things and stuff, as well as the relationship between pairs of things are incorporated. Such relationships are highlighted with the arrows this this figure. Our model achieves the improvement on both detection and segmentation task as described in Chap. II.

images have been proposed [26, 106]. These techniques find correspondences between images by matching consistent features. 3D structure and camera motion can be reconstructed using trajectories of the imag features.

Understanding geometry from a single 2D RGB image is another field of interest in computer vision. Different methods are proposed by using 3D indoor layout priors [59, 85], or 3D primitive shape as priors for objects in the scene [53, 57]. The context among objects or between objects and layout has also been studied recently [22, 60].

1.3 Our Contribution

1.3.1 A model for understanding *things and stuff*

We propose a framework for scene understanding that models both things and stuff using a joint representation (Fig. 1.2). This representation allows us to enforce sophisticated geometric and semantic relationships between thing and stuff categories in a single graphical model. We use the latest advances in the field of discrete optimization to efficiently perform maximum a posteriori (MAP) inference in this model. We

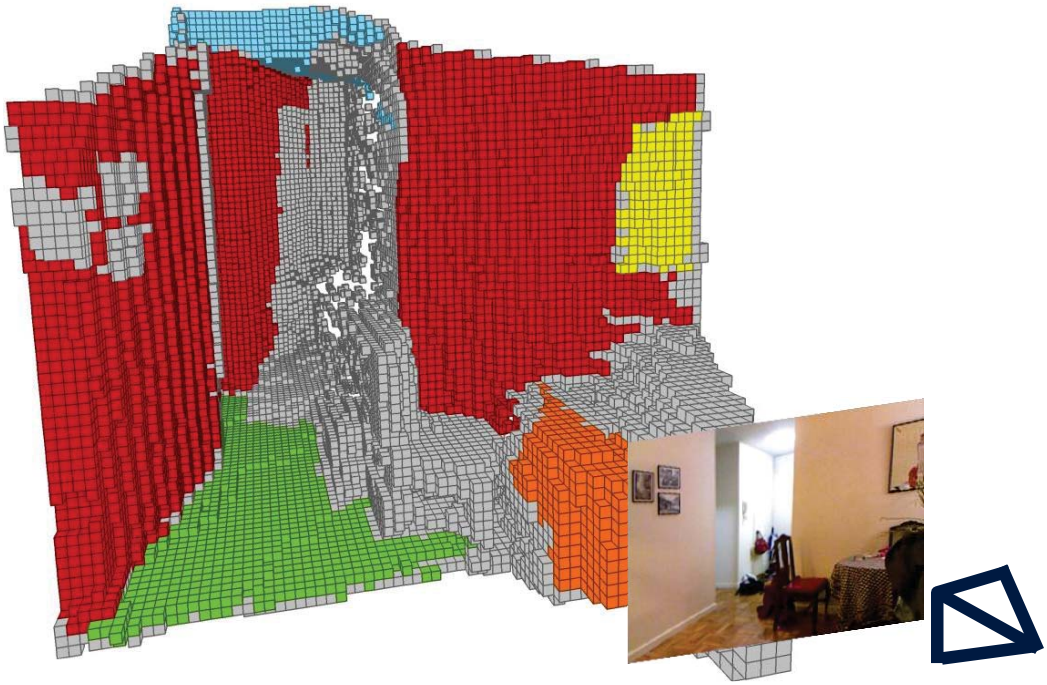


Figure 1.3: The VCRF model (Chap. III) finds the best configuration of semantic labels and geometric properties for 3D space represented by a set of voxels. In the figure, red corresponds to 'wall', green to 'floor', orange to 'table' and yellow to 'picture'.

evaluate our method by comparing it to state-of-the-art methods for object segmentation and detection on public datasets such as Stanford dataset and the challenging PASCAL'09 segmentation dataset. This framework is described in Chap. II.

1.3.2 A Voxel-CRF based model

In the past few years, researchers have taken advantage of the recent diffusion of depth-RGB (RGB-D) cameras to help simplify the problem of inferring scene semantics. However, while the added 3D geometry is certainly useful for segmenting out objects with different depth values, it also adds complications. The 3D geometry is often incorrect because of noisy depth measurements, and the actual 3D extent of the objects is usually unknown because of occlusions. We propose a new method that

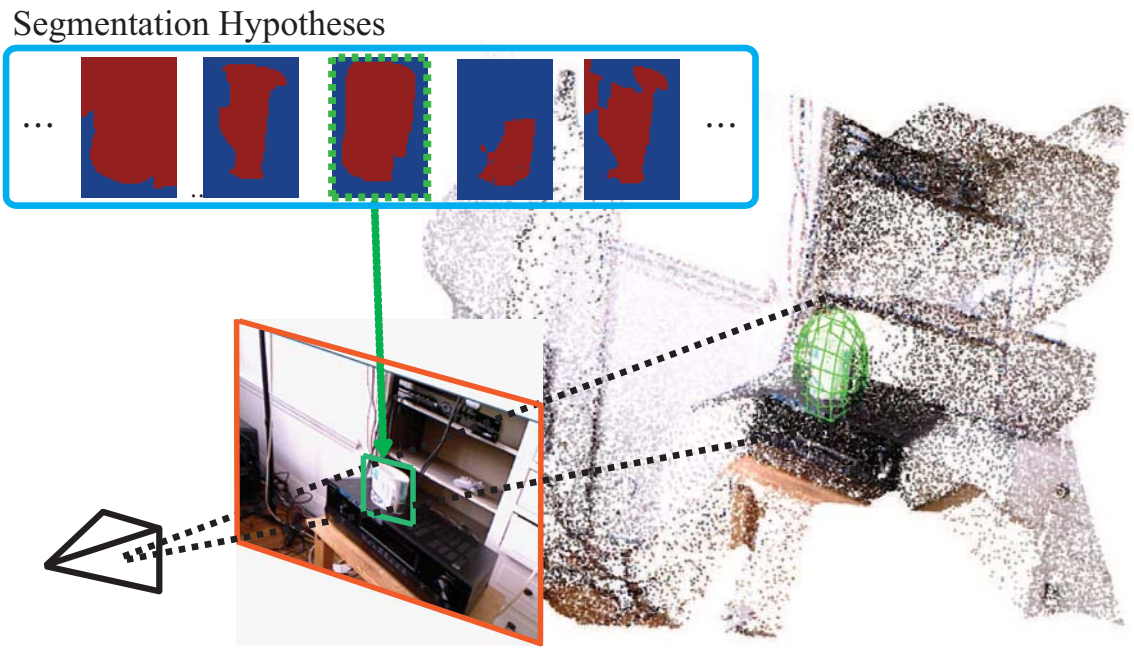


Figure 1.4: Find an accurate 3D location of object from RGB-D data is challenging task. We propose a new framework to obtain accurate localizations of objects in 3D by exploring segmentation hypotheses of the object in 2D.

allows us to jointly refine the 3D reconstruction of the scene (raw depth values) while accurately segmenting out the objects or scene elements from the 3D reconstruction (Fig. 1.3). This is achieved by introducing a new model which we called Voxel-CRF. The Voxel-CRF model is based on the idea of constructing a conditional random field over a 3D volume of interest which captures the semantic and 3D geometric relationships among different elements (voxels) of the scene. This model allows us to jointly estimate (1) a dense voxel-based 3D reconstruction and (2) the semantic labels associated with each voxel even in the presence of partial occlusions using an approximate yet efficient inference strategy. We evaluated our method on the challenging NYU Depth dataset (Version 1 and 2). Our method is described in Chap. III.

1.3.3 A model for understanding things in 3D

We focus on the problem of finding objects in 3D from RGB-D images, and propose a novel framework that explores the compatibility between segmentation hypotheses of an object in the image and the corresponding 3D map (Fig. 1.4). Our framework allows us to discover the optimal location of the object in 3D using a generalization of the structural latent SVM formulation in 3D as well as the definition of a new loss function defined over 3D space which penalize the amount of incorrect 3D localization during training. We evaluate our method using two existing RGB-D datasets. Extensive quantitative and qualitative experimental results show that our proposed approach outperforms the state-of-the-art as well as a number of baseline methods for both 3D and 2D object recognition tasks. Our framework is described in Chap. IV.

1.3.4 A model for understanding things in 2D

Using semantic hierarchies of object labels for visual categorization has been shown to have a number of important benefits. The hierarchies allow us to recognize objects at different levels of semantic granularity. Moreover, they enable significant gains in efficiency (e.g., logarithmic with the number of categories [48, 93]) or the construction of a more meaningful distance metric for image classification [98]. A critical question, however, still remains controversial: would structuring data in a hierarchical sense also help classification accuracy? In this work we address this question and show that a hierarchically structured database can be indeed successfully used to enhance classification accuracy using a sparse approximation framework. We propose a new formulation for sparse approximation where the goal is to discover the sparsest path within a hierarchical data structure that best represents the query object. Extensive quantitative and qualitative experimental evaluation on a number of branches of the Imagenet database [27] as well as on the Caltech-256 [48] demonstrate our theoretical claims and show that our approach produces better hierarchical categorization results

than competing techniques. This approach is described in Chap. V.

1.3.5 Themes

In this thesis, the following themes appear throughout many chapters.

Graphical model. We solve the scene understanding problem by representing the 2D/3D space with graphical models, where each vertex represents image elements and edges connect adjacent elements. Depending on the problem, variations of conditional random fields (CRF) model are proposed as in Chap. II (ACRF) and Chap. III (VCRF). Specifically, ACRFs model the things and stuff using 2D image representations, where each image element represents a pixel or a super-pixel. On the other hand, VCRFs model voxels in 3D space.

Training strategies. The complexity of the labeling space for the scene understanding problem makes the training process highly challenging. We adopt state-of-the-art machine learning techniques such as structural support vector machine [121] to learn linear weights balancing the importance of different factors. The training methods proposed in this thesis are discriminative approaches in the presence of a training set. In other words, we have assumed that testing and training data have similar properties. Notice however, that during the training process, training parameters are being regularized so that the model can avoid severe bias toward a training set.

Inference strategies. The proposed CRF models are built on top of highly connected graph structures. While this allows us to handle richer interactions among image elements, it makes the inference task challenging. We propose several methods to efficiently solve the complex inference problems. For example, in Chap. II, we propose to approximate the energy function so that the optimization can be efficiently performed using a graph-cut method while preserving the model's semantic and geometric properties. In Chap. II, iterative approaches are introduced to efficiently infer

the best configuration of random variables representing semantic labels and geometric status.

Experimental evaluation. In order to evaluate our models and compare them with state-of-the-art baseline methods, we conduct experiments on public available dataset such as Stanford dataset [44], PASCAL dataset [30], and NYU Depth dataset [97, 111]. When necessary, we augmented existing dataset with additional labels; for example, pixelwise annotation for thing classes are added to the Stanford dataset. For the NYU Depth dataset, we created top-view semantic floor map as described in Chap. III.

CHAPTER II

Relating Things and Stuff via Object Property Interactions

In the last few years, substantially different approaches have been adopted for segmenting and detecting "things" (object categories that have a well defined shape such as people and cars) and "stuff" (object categories which have an amorphous spatial extent such as grass and sky). While things have been typically detected by sliding window or Hough transform based methods, detection of stuff is generally formulated as a pixel or segment-wise classification problem. This work proposes a framework for scene understanding that models both things and stuff using a common representation but still preserves their distinct nature by using a property list. This representation allows us to enforce sophisticated geometric and semantic relationships between thing and stuff categories in a single graphical model. We use the latest advances made in the field of discrete optimization to efficiently perform maximum a posteriori (MAP) inference in this model. We evaluate our method on the Stanford dataset by comparing it against state-of-the-art methods for object segmentation and detection. We also show that our method achieves competitive performances on the challenging PASCAL'09 segmentation dataset.

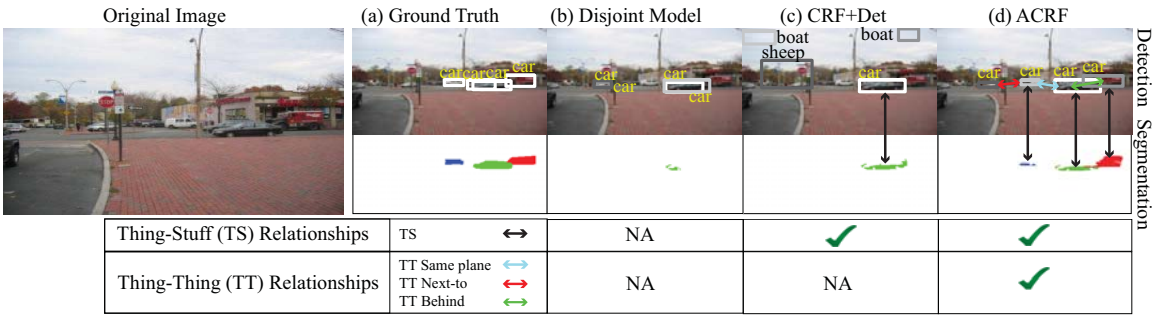


Figure 2.1: Our goal is to segment the image into things (e.g., cars, humans, etc) and stuff (e.g., road, sky, etc) by combining segmentation (bottom) with object detection (top). Results from different variants of our method (capturing a subset of critical contextual relationships) are shown from left to right columns. At the top of each column, we show the top 4 probable bounding boxes, where light and dark boxes denote the confidence ranking from high to low. Instance-based segmentation are shown in each bottom column, where different colors represent different object instances. Notice that our final ACRF captures the key relationships and recovers many missing detections and segmentation labels. *Thing-Stuff* and *Thing-Thing* relationships are indicated by color-coded arrows connecting pairs of things. Different color codes indicate different types of relationships.

2.1 Statement

A number of methods have been proposed for recognizing *things* and *stuff*. For detecting *things*, we find that methods based on pictorial structures (i.e., Felzenszwalb et al. [35]), pyramid structures (i.e., Grauman and Darrell [47]), generalized Hough transform [5, 41, 87, 92, 116], or multi-view model [114, 127] work best. These representations are appropriate for capturing shape or structural properties of *things*, and typically identify the object by a bounding box. The second category of methods aim at segmenting the image into semantically consistent regions [58, 72, 110] and work well for *stuff*, like sky or road.

In order to coherently interpret the depicted scene, various types of contextual relationships among objects (stuff or things) have been explored. For example, co-occurrence relationships (e.g., cow and grass typically occur in the same image) have been captured in [80, 102], 2D geometric relationships (e.g., below, next-to, etc) have been utilized in [28, 52, 62], 2.5D geometry relationships (e.g., horizon line) have

been incorporated by Hoiem et al. [65] and Bao et al. [3]. The use of such contextual relationships have inspired the development of robust algorithms for various object recognition tasks. For instance, many segmentation methods [44, 80, 125] have been proposed to capture stuff-stuff relationships in a random field formulation. Similarly, thing-thing relationships have been incorporated into a random field for jointly detecting multiple objects (Desai et al. [28]).

Recently, researchers have proposed methods to jointly detect *things* and segment *stuff*. Gould et al. [45] proposed a random field model incorporating both stuff-stuff, thing-stuff, and thing-horizon relationships. One limitation of their approach is that it cannot capture 2D geometric and co-occurrence relationships between things. Moreover, inference is computationally very demanding and typically takes around five minutes per image. To overcome this limitation, some authors have proposed inference procedures which leverage existing approaches for detection and segmentation and use the output of such approaches as input features in an iterative fashion [63, 66, 115]. Unfortunately, optimality is not guaranteed for most of these approaches.

We propose a novel framework for jointly detecting things and segmenting stuff that, for the first time, can coherently capture many known types of contextual relations between thing-thing, stuff-stuff, and thing-stuff categories. Our contributions are three-fold. First, the model infers both geometric and semantic relationships describing the objects (i.e., object x is behind object y) via object property interactions. Second, the model enables instance based segmentation (see color coded segments in Fig. 2.1(d)) by associating segments of thing categories to instance-specific labels. Finally, the special design of model potentials allows efficient inference which takes a few seconds per image in average and is performed using a combination of state-of-the-art discrete optimization techniques.

Object hypothesis and property lists. Our framework extends the basic con-

ditional random field (CRF) formulations for scene segmentation (i.e., stuff recognition) [72, 110] by introducing the concept of a generic object instance hypothesis. Every object instance hypothesis is described by a *property list* (Fig. 2.2-Top). This list includes semantic properties, such as the object categorical label l. Further, depending on the category label (stuff or thing), the hypothesis can also be characterized by some geometric properties, such as the 2D location (u, v), and the distance from the camera (depth) d.

We augment the above-mentioned CRF formulation with object hypothesis indicator variables which capture the presence or absence of an object hypothesis (see Fig. 2.2(a)-Top). We refer to our model as the augmented CRF, or ACRF, to highlight the newly added object hypothesis indicator variables. The indicator variables can take only two states: 0 or 1 which represents the absence or presence of an hypothesis, respectively. The key benefit of the object indicator variables is that they allow us to easily encode sophisticated semantic and geometric relationships between pairs of object hypotheses. For instance, simple pairwise potentials defined over object indicator variables can allow to incorporate i) 2D geometric relationships such as "above" which model the property that one hypothesis lies above the other (e.g., a person sitting on a bike), ii) depth and occlusion relationships such as "in-front" which model the property that one hypothesis lies in front of the other (e.g., a person standing in front of a car), and iii) support relationships which model the property that one hypothesis is supported by another hypothesis (e.g., pedestrians walking on a road). More sophisticated relationships such as a composition of these basic 2D or 2.5D relationships can also be supported. Critically, the ACRF model generalizes Ladicky et al.'s model [80] which captures stuff-stuff co-occurrence contextual relationships only. In contrast, our model can not only encode relationships between stuff categories that depend on their geometrical properties such as (orientation, depth) but can also encode geometrical and semantic properties between stuff and things as

well as things and things. Our model can also handle multiple instances of thing object categories and, thus, also generalizes the work of Barinova et al. [5]. We illustrate the efficacy of our approach in Fig. 2.1. As seen in the figure, detections typically do not agree with the segmentation results (Fig. 2.1(b)) if the detection and segmentation are applied separately. A model capturing thing-stuff relationships ensures consistency between detection and segmentation results (Fig. 2.1(c)). Finally, when thing-thing relationships (e.g., next-to, behind, same plane, etc) are included, even small objects, that are hard to detect and segment, can be discovered (Fig. 2.1(d)).

Learning. Relationships encoded by our model are regulated by a number of model parameters that we learn from training data. The relationships can be both attractive (e.g., a person is likely to sit on a motorbike) and repulsive (e.g., car and airplane are unlikely to co-occur), and are enforced by adding positive or negative costs to the energy of the model. We formulate the problem of learning these costs as a Structured SVM (SSVM) [121] learning problem with two types of loss functions related to the segmentation loss and detection loss, respectively (see Sec. 2.5 for details).

MAP Inference. Jointly estimating the segmentation variables X and object indicator variables Y (Fig. 2.2(c)) is challenging due to the intrinsic difference of the variable space and the complex types of pair-wise relationships between thing-thing, and thing-stuff. We design an efficient graph-cut-based move making algorithm by combining state-of-the-art discrete optimization techniques. Our method is based on the α-expansion move making approach [17], which works by projecting the energy minimization problem of segmentation variables X into a binary energy minimization problem to have the same space as the indicator variables Y. We use the "probing" approach similar to the one described by Rother et al. [104] to handle the non-submodular function related to pair-wise object relationships (i.e., thing-thing). Our MAP inference algorithm takes only a few seconds per image in average as opposed to five minutes by Gould et al. [45].

2.2 Related Work

Our method is closely related to the following three formulations which all can be considered as special cases of our model. Desai et al. [28] propose a CRF model capturing thing-thing relationships and show that object detection performance can be consistently improved for multiple object categories. Their model can be considered as a special case of our model when no segmentation variable X exists. Both Ladicky et al.'s methods [80, 81] extend the standard CRF model to incorporate more sophisticated relationships. Ladicky et al. [81] incorporate thing-stuff relationships and demonstrate that the information from object detection can be used to improve the segmentation performance consistently across all object categories. Their model can be considered as a special case of our model when no thing-thing relationship is incorporated. Ladicky et al. [80] propose to capture co-occurrence types of object relationships and demonstrate that the co-occurrence information can be used to improve the segmentation performance significantly. Their model can also be considered as a special case of our model when no thing-thing relationships are established and no geometric property interactions (i.e., above, same horizon, etc) are introduced. Finally, [80] cannot be used to assign segments to object instances or localize object instances. Our results on the Stanford dataset demonstrate that our model achieves superior performances than [80, 81].

The Semantic Structure From Motion (SSFM) proposed by Bao et al. [2] also jointly model object instances and regions. However, unlike our method which utilizes one single image, their approach utilizes the correspondences of object instances and regions established across multiple images to improve the object detection and segmentation performance. Li et al. [88] proposes a uniform model to jointly classify the scene, recognize the class of each segment, and annotate the image with a list of tags. However, the model cannot localize each object instances. Hence, the thing-thing and thing-stuff interactions are not incorporated in the model.

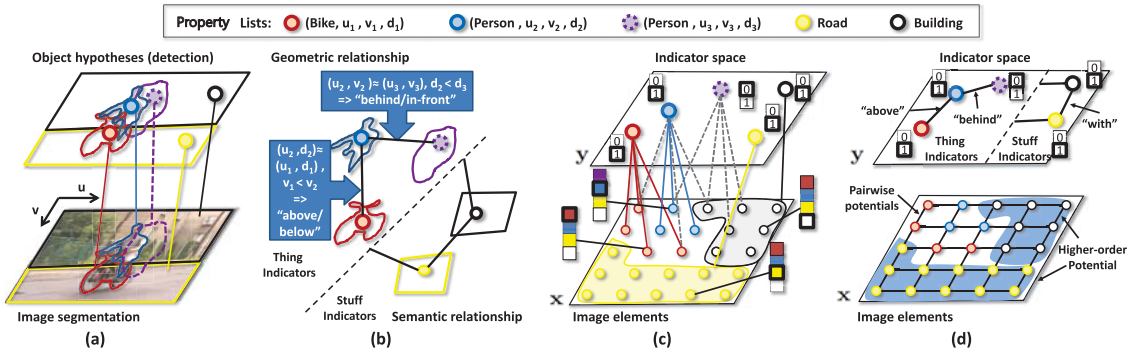


Figure 2.2: Our Augmented CRF model (ACRF). In panel (a), we show an image and the indicator variables corresponding to the different object hypotheses present in it. The instance hypotheses for 'thing' categories such as person and bike have geometric properties (e.g., spatial and depth). These properties are absent for stuff categories such as building and road. In panel (b), we demonstrate how relationships (e.g., behind, above, etc) are selected given a pairs of property lists. In panel (c), the figure shows the label space of the segmentation variables $\mathbf{X}$ and the indicator variables $\mathbf{Y}$ and the interaction between them. In panel (d)-Bottom, the figure shows the pairwise and higher order interactions among segmentation variables $\mathbf{X}$ which are present in standard CRF formulations. In panel (d)-Top, the figure shows the pairwise interactions among indicator variables $\mathbf{Y}$ which can encode different geometric and semantic relationships. The two edges connected to the stuff indicators which end in the dashed separator line indicate that stuff indicators are interacting with thing indicators with only category property.

2.3 Augmented CRF

We now explain our Augmented Conditional Random Field (ACRF) model. ACRF jointly models object detection and segmentation (Fig. 2.2 (a)), and can incorporate contextual relationships between things and stuff, and between multiple things (Fig. 2.2 (b)).

Object segmentation, like other image labeling problems, is commonly formulated using Conditional Random Fields (CRF). The conventional CRF model is defined over a set of random variables $X = \{x_i\},\ i \in \mathcal{V}$ where $\mathcal{V}$ represents the set of image elements, which could be pixels, patches, super-pixels, etc (Fig. 2.2 (c)-Bottom). Each random variable x_i is assigned to a label from a discrete label space $\mathbf{L}$, which for the task of object-category segmentation, is considered the set $\mathcal{L}$ of object categories such

as grass, road, car and people.

The energy (or cost) function $E(X)$ of the CRF is the negative logarithm of the joint posterior distribution of the model and has the following form: $E(X) = -\log P(X|\mathcal{E}) = -\log \phi_{eRF}(X|\mathcal{E}) + \mathrm{K} = \sum_{c \in \mathcal{C}^X} \psi_c(X_c) + \mathrm{K}$, where $\mathcal{E}$ is the given evidence from the image and any additional information (e.g., object property lists), $\phi_{eRF}(X|\mathcal{E})$ takes the form of a higher order CRF model defined over image elements (Fig. 2.2 (d)-Bottom). $\phi_{eRF}(X|\mathcal{E})$ can be decomposed into potential ψ_c which is a cost function defined over a set of element variables X_c (called a clique) indexed by $c \in \mathcal{C}^X$; $\mathcal{C}^X$ is the set of cliques for image elements, and K is a constant related to the partition function. The problem of finding the most probable or maximum a posteriori (MAP) assignment of the CRF model is equivalent to solving the following discrete optimization problem: $X^* = \arg\min_{X \in \mathcal{L}^{|\mathcal{V}|}} E(X)$.

The standard CRF model mostly relies on bottom-up information. It is constructed using unary potentials based on local classifiers and smoothness potentials defined over pairs of neighboring pixels. Higher-order potentials such as the ones used in [72] encourage labels of groups of image elements to be the same. This classic representation for object segmentation has led to excellent results for the stuff object categories, but has failed to replicate the same level of performance on the thing object categories. The reason for this dichotomy lies in the model's inability to explicitly encode the relationship between the shape and relative positions of different parts of structured objects categories such as the head and the torso of a person.

Part-based models such as Pictorial Structures [37], Latent SVM (LSVM) [35], and Hough transform based models [5, 87] have shown to be much more effective at detecting 'things'. One of our contributions is proposing a unified framework to incorporate all instance hypotheses from these methods as additional object-instance evidences. In our model, every piece of object-instance evidence is characterized by the property lists. These properties include the category of the object l_j, the spatial

location in the image (u_j, v_j) at which the object is seen, and the depth or distance d_j of the object instance from the camera.

In addition to the variables representing image elements, our model contains a set of indicator variables (later referred as indicators) $Y = \{y_j \in \{0, 1\}\}$ for every possible configuration $j \in \hat{\mathcal{Q}}$ of an object (Fig. 2.2 (c)-Top). The configuration set $\hat{\mathcal{Q}}$ is a Cartesian product of the space of all possible object category labels $\mathcal{L}$, all possible spatial locations $U \times V$ in the image, and all depth or distance values within a range $[0, D]$. For example, a configuration $j \in \hat{\mathcal{Q}}$ specifies that an instance of the object category $l_j \in \mathcal{L}$ exists at location (u_j, v_j) in the image at a distance d_j away from the observer. We also associate with each object instance a segmentation mask $\mathcal{V}_j$ which is the set of image elements associated with the object. In order to handle uncertainty in location and distance from the camera for stuff (e.g., a grass region may have a large spatial extent and it may be at a range of distances from the camera) and thing categories which are not detected, we allow a configuration j including general hypothesis specified by l_j only (i.e., without specifying the location of the hypothesis).

As mentioned earlier, variables X representing the image elements in the classical CRF formulation for object segmentation take values from the set of object categories $\mathcal{L}$ only. In contrast, in our framework, these variables take values from a set of all possible object configuration $x_i \in \mathbf{L} = \hat{\mathcal{Q}}$ (refer as *augmented labeling space*). On the one hand, this allows us to obtain segmentations of individual instances of particular object categories which the classical CRF formulations are unable to handle. On the other hand, the space $\hat{\mathcal{Q}}$ of all possible detections is clearly huge, which makes learning and inference much more challenging. We will come back to this issue later.

The joint posterior distribution of the segmentation X and indicator variables Y can be written as:$P(X, Y|\mathcal{E}) \propto \phi_{eRF}(X|\mathcal{E})\ \phi_{oRF}(Y|\mathcal{E})\ \phi_{con}(X, Y|\mathcal{E})$. The functions ϕ_{oRF} take the form of a CRF model defined over object indicator variables as follows:

$\phi_{oRF}(Y|\mathcal{E}) = \prod_{c \in \mathcal{C}^Y} e^{\varphi_c(Y_c)}$, where the potential $\varphi_c(Y_c)$ is a cost function defined over a set of indicator variables Y_c indexed by $c \in \mathcal{C}^Y$, and $\mathcal{C}^Y$ is the set of cliques of indicators. The potential function ϕ_{con} enforces that the segmentation and indicator variables take values which are consistent with each other (Fig. 2.2 (c)). The term is formally defined as: $\phi_{con}(X, Y|\mathcal{E}) = \prod_{j \in \hat{\mathcal{Q}}} e^{\Phi(y_j, X)}$, where $\Phi(y_j, X)$ is the potential relating each indicator y_j with a specific set of elements $\mathcal{V}_j$ in X. Hence, the model energy can be written as:

$$E(X, Y) = \sum_{c \in \mathcal{C}^X} \psi_c(X_c) + \sum_{j \in \hat{\mathcal{Q}}} \Phi(y_j, X) \tag{2.1}$$

$$+ \sum_{c \in \mathcal{C}^Y} \varphi_c(Y_c) \ . \tag{2.2}$$

The first term of the energy function is defined in a manner similar to [72]. We now describe other terms of the energy function in detail in the following subsections.

Implicit representation of inactive object configurations. It is easy to see that the space $\hat{\mathcal{Q}}$ of all possible configurations is huge, which would make learning and performing inference in the above model completely infeasible. However, in real world images, only a few possible configurations are actually present. Thus, most indicator variables $y_j,\ j \in \hat{\mathcal{Q}}$ are inactive (take value 0), and similarly the label set for the segmentation variables is typically quite small. We use an object detector that has been trained on achieving high recall rate to generate the set of *plausible* object configurations $\mathcal{Q}$ instances that are likely to be present in any given image. In this way, we reduce the problem into a manageable size for the inference algorithm.

2.3.1 Relating Y and X

The function $\Phi(y_j, X)$ (Fig. 2.2(c)) is a likelihood term that enforces consistency in the assignments of the jth indicator variable y_j and a set of segmentation variables

X. It is formally defined as:

$$\Phi(y_j, X) = \begin{cases} \inf & \text{if } y_j \neq \delta_j(X) \\ \gamma_{l_j} \cdot |\mathcal{V}_j| & \text{if } y_j = \delta_j(X) = 1 \\ 0 & \text{if } y_j = \delta_j(X) = 0 \end{cases}, \tag{2.3}$$

where j is any possible object configuration in $\mathcal{Q}$, the function $\delta_j(X)$ indicates whether the indicator j shares a consistent object category label with image elements in $\mathcal{V}_j$, and is defined as:

$$\delta_j(X) = \begin{cases} 1 & \text{if } R_j(X) = \frac{|\mathcal{V}_j(X)|}{|\mathcal{V}_j|} \geq R(l_j) \\ 0 & \text{otherwise} \end{cases}, \tag{2.4}$$

where $|\mathcal{V}_j(X)| = |\{i; x_i = l_j \texttt{ for } i \in \mathcal{V}_j\}|$ is the number of elements in $\mathcal{V}_j$ assigned with label l_j, $|\mathcal{V}_j|$ is the total number of elements in $\mathcal{V}_j$, $R_j(X)$ is the consistency percentage, and $R(l_j) \in [0\ 1]$ is an object category-specific consistency threshold. Hence, the first condition in the above function ensures that $y_j = 1$ if and only if the detection j shares an object label with at least $R(l_j)$ percent of the pixels (or image element) in $\mathcal{V}_j$ (i.e. $R_j(X) \geq R(l_j)$). The remaining conditions in Eq. 2.3 shows that this potential is an Occam razor or MDL prior, similar to [5, 80] so that the model is penalized by $\gamma_{l_j} \cdot |\mathcal{V}_j|$ when $y_j = 1$.

2.3.2 Object Indicator CRF

The object indicator CRF potential $\varphi_c(Y_c)$ in Eq. 2.2 can be decomposed into two terms as follows, $\sum_{c \in \mathcal{C}^Y} \varphi_c(Y_c) = \sum_{j \in \mathcal{Q}_1} \varphi_u(y_j) + \sum_{(j,k) \in \mathcal{U}} \varphi_p(y_j, y_k)$, where $\mathcal{Q}_1 \subset \mathcal{Q}$ is the set of thing indicators with geometric properties and $\mathcal{U}$ is the set of pairs of indicators, which interact with each other.

The term $\varphi_u(y_j)$ is the unary potential for the thing indicator, defined as:

$$\varphi_u(y_j) = \begin{cases} \beta_j \cdot |\mathcal{V}_j|, \ \texttt{if} \ \ y_j = 0 \\ 0, \ \texttt{if} \ \ y_j = 1 \end{cases} , \tag{2.5}$$

such that the cost of suppressing hypothesis j (i.e., label y_j as 0) is $\beta_j \cdot |\mathcal{V}_j|$ (proportional to the detection confidence).

The term $\varphi_p(y_j, y_k)$ (black edges in Fig. 2.2 (d)-Top) represents the interactions between any pair of indicator variables. Depending on the types of properties associated with the pair of indicator variables, this term can represent a number of relationships. It can not only model spatial relationship in 2D such as the ones learned and employed in the approach proposed by [28], but also model behind and in-front relationships given the depth property. The term can also encode co-occurrence relationships [80] for pairs of indicators with only category properties.

For any pair of indicators $j, k \ \in \mathcal{Q}$, the term is formally defined as:

$$\varphi_p(y_j, y_k) = w_{l_j,l_k}^{r_{jk}}(y_j, y_k) \cdot \max(|\mathcal{V}_j|, |\mathcal{V}_k|) \ , \tag{2.6}$$

where r_{jk} is the type of relationship that we want to enforce between the pair of object instances j and k, and is a subset of the overall relationship set $\mathcal{R}$, which is defined as: $\mathcal{R} = \{$*co-occur, above, below, next-to, in-front, behind, or the composition of them*$\}$.

The pseudo-boolean function $w_{l_j,l_k}^{r_{jk}}(y_j, y_k) : \{0,1\}^2 \rightarrow \mathbb{R}$ specifies the cost of all 4 possible joint assignments of y_j and y_k under the relationship r_{jk} for a pair of object categories l_j, l_k. As a result, the potential can capture both attractive (i.e., $w(0,0)+w(1,1) \leq w(0,1)+w(1,0)$) and repulsive interactions (i.e., $w(0,0)+w(1,1) \geq w(0,1)+w(1,0)$). For example, a person usually is sitting on a motorbike (attractive), and cars do not overlap with each other in 3D (repulsive).

The relationship r_{jk} is specified by the properties associated with the detection instances j and k. For instance, if the object indicators i and j have only object category properties, the relationship r_{jk} models the co-occurrence cost of the object categories. In this case, we assume

$$w^{co}_{j,k}(y_j, y_k) = \begin{cases} \gamma_{l_j,l_k} & \text{if } y_j = y_k = 1 \\ 0 & \text{otherwise} \end{cases}, \tag{2.7}$$

where γ_{l_j,l_k} is the co-occurrence cost for object categories l_j and l_k. From the above definition, it is easy to see that our model generalizes both CRF models proposed in [28, 80].

2.4 Inference

We now show that the MAP inference problem in our ACRF model can be solved by minimizing the energy function using an efficient graph cut based expansion move making algorithm [17].

Standard move making algorithms repeatedly project the energy minimization problem into a smaller subspace in which a sub-problem is efficiently solvable. Solving this sub-problem produces a change to the solution (referred to as a move) which results in a solution having lower or equal energy. The *optimal* move leads to the largest possible decrease in the energy.

The *expansion* move algorithm projects the problem into a Boolean label sub-problem. In an α-expansion move, every segmentation variable X can either retain its current label or transit to the label α. One iteration of the algorithm involves making moves for all α in $\mathcal{L}$ successively. Under the assumption that the projection of the energy is pairwise and submodular, it can be exactly solved using graph cuts [12, 73]. We derive graph construction only for energy terms related to indicator variables Y,

for all other terms, the constructions are introduced in [17, 72].

2.4.1 Functions of indicator variables Y with only category property.

The energy terms related to the indicator variables, whose only property is a category label, are $\Phi(y_j, X)$ in Eq. 2.3 and $\varphi_p(y_j, y_k)$ in Eq. 2.6 and 2.7. Observe that we can represent the combination of these terms as a function, $\mathcal{F} : L \subset \mathcal{L} \rightarrow \mathbb{R}$ as

$$\mathcal{F}(L(Y)) = \min_{X} \sum_{j \in \mathcal{Q}_2} \Phi(y_j, X) + \sum_{(j,k) \in \mathcal{U}_2} \varphi_p(y_j, y_k) \ , \tag{2.8}$$

where $L(Y) = \{l_j; k \in \mathcal{Q}_2, y_j = 1\}$ is a set of existing object categories (i.e., $y_j = 1$), $\mathcal{Q}_2$ is any subset of the indicator variables, whose only property is a category label, and $\mathcal{U}_2$ is a subset of $\mathcal{U}$ such that $j, k \in \mathcal{Q}_2$. From the definition of the term in section 2.3.1 and 2.3.2, we can see that $\mathcal{F}(\{l_j\}) = \gamma_{l_j}|\mathcal{V}_j|$. Furthermore, $\mathcal{F}(\{l_j, l_k\}) = \mathcal{F}(\{l_j\}) + \gamma_{l_k}|\mathcal{V}_k| + \gamma_{l_j,l_k}$; $\mathcal{F}(\{l_j, l_k, l_q\}) = \mathcal{F}(\{l_j, l_k\}) + \gamma_{l_p}|\mathcal{V}_p| + \gamma_{l_j,l_p} + \gamma_{l_k,l_q}$. It can be easily seen that the above function satisfies the properties of the co-occurrence potential:

$$L_1 \subset L_2 \Longrightarrow \mathcal{F}(L_1) \leq \mathcal{F}(L_2) \ , \tag{2.9}$$

proposed by [80] allowing us to use their graph construction for minimizing this energy function.

2.4.2 Functions of indicator variables Y with instance properties.

The energy terms related to the instance indicator variables are $\Phi(y_j, X)$ in Eq. 2.3 and $\varphi_p(y_j, y_k)$ in Eq. 2.6. Since $\varphi_p(y_j, y_k)$ in Eq. 2.6 captures both repulsive and attractive pair-wise relationships, it can not be combined with $\Phi(y_j, X)$ in Eq. 2.3 to form a co-occurrence potential satisfying Eq. 2.8. However, $\Phi(y_j, X)$ can be approx-

imated as:

$$\Phi(y_j, X) = \gamma_j(y_j \frac{1 - R_j(X)}{1 - R(l_j)} + (1 - y_j)\frac{R_j(X)}{R(l_j)}) \ . \tag{2.10}$$

The detailed derivation and the corresponding α-expansion move energy for this term is described in the Sec. 2.4.3.

The graph construction of the pair-wise instance indicators in Eq. 2.6 is equivalent to the construction of binary variables which is clearly described in [17]. However, since we would like to capture both attractive (i.e., both indicators having the same labels) and repulsive (i.e., both indicators having different labels) interactions, some functions could be submodular and some could be non-submodular in (y_j, y_k), respectively. Since each indicator only interacts with other nearby indicators, a simple "probing" approach similar to the one described in [104] can effectively handle the non-submodular function related to pair-wise object-instance interaction. As a result, our inference algorithm does not rely on sophisticated techniques such as QPBO [104] which requires more memory and computation time.

2.4.3 Functions of Thing Indicators Y

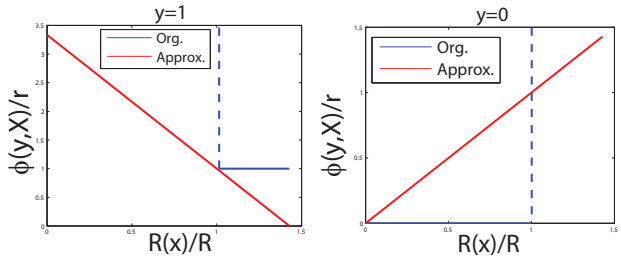


Figure 2.3: Comparison between the original function $\Phi(y, X)$ (blue line) and the approximated function (red lines) in Eq. 2.12 and 2.11. The left panel shows the case when $y = 1$. The right panel shows the case when $y = 0$. Notice the dash blue lines indicate the sharp transition from finite values to infinite values.

We observe in Eq. 2.3, when $y_j = 1$

$$\Phi(y_j, X) = \begin{cases} \inf & \text{if } \delta_j(X) = 0 \\ \gamma_j & \text{if } \delta_j(X) = 1 \end{cases} \approx \gamma_j \frac{1 - R_j(X)}{1 - R(l_j)} \ . \tag{2.11}$$

When $y_j = 0$

$$\Phi(y_j, X) = \begin{cases} \inf & \text{if } \delta_j(X) = 1 \\ 0 & \text{if } \delta_j(X) = 0 \end{cases} \approx \gamma_j \frac{R_j(X)}{R(l_j)} \ . \tag{2.12}$$

Hence, $\Phi(y_j, X)$ becomes,

$$\Phi(y_j, X) = \gamma_j (y_j \frac{1 - R_j(X)}{1 - R(l_j)} + (1 - y_j) \frac{R_j(X)}{R(l_j)}) \ . \tag{2.13}$$

The effect of the approximation in Eq. 2.12 and 2.11 are shown in Fig. 2.3. Instead of imposing an infinite cost when $\delta(X) \neq y$, our approximation imposes an cost which is linearly proportional to the consistency percentage $R(X)$. When $y = 1$, the ratio between the consistency percentage and the consistency threshold $R(X)/R(l)$ are encouraged to be large, which means the more elements in X are labeled as l, the better (Fig. 2.3-Left). On the contrary, when $y = 0$, the ratio between the consistency percentage and the consistency threshold $R(X)/R(l)$ is encouraged to be small, which means the less elements in X are labeled as l, the better (Fig. 2.3-Right).

2.4.3.1 α-expansion move energy

We first define the transformation function $T_\alpha(x_i; t_i)$ for the α-expansion move which transforms the label of a random variable x_i as:

$$T_\alpha(x_i, t_i) = \begin{cases} \alpha, & \text{if } t_i = 0 \\ x_i, & \text{if } t_i = 1 \end{cases} \tag{2.14}$$

The corresponding α-expansion move energy for the term in Eq. 2.13 can be written as: $\Phi(y_j, T_\alpha(X, T)) =$

$$\begin{cases} \gamma_j(\frac{y_j}{1-R(l_j)}(1 - R_j(X) + \sum_{i \in \mathcal{V}_j(X)} \frac{(1-t_i)}{|\mathcal{V}_j|}) \\ +\frac{1-y_j}{R(l_j)}(\sum_{i \in \mathcal{V}_j(X)} \frac{(t_i)}{|\mathcal{V}_j|})), \text{if } \alpha \neq l_j \\ \gamma_j(\frac{1-y_j}{R(l_j)}(R_j(X) + \sum_{i \in \mathcal{V}_j \setminus \mathcal{V}_j(X)} \frac{(1-t_i)}{|\mathcal{V}_j|}) \\ +\frac{y_j}{1-R(l_j)}(\sum_{i \in \mathcal{V}_j \setminus \mathcal{V}_j(X)} \frac{(t_i)}{|\mathcal{V}_j|})), \text{if } \alpha = l_j \end{cases} \tag{2.15}$$

where $\mathcal{V}_j \setminus \mathcal{V}_j(X)$ is the remaining set of elements in $\mathcal{V}_j$ with labels (i.e.,$\{x_i \neq l_j; i \in \mathcal{V}_j\}$). Notice that when $\alpha \neq l_j$ the function is submodular in (y_j, t_i), but when $\alpha = l_j$ it is submodular in $(\overline{y}_j, t_i)$, where $\overline{y}_j = 1 - y_j$ is the negation of y_j.

After the transformation, the first two terms of the original model energy (Eq. 2.2) becomes a pairwise and submodular function of T, Y, and $\overline{Y}$ as follows,

$$E(T, Y, \overline{Y}) = \sum_{c \in \mathcal{C}^X} \psi_c(T_c) + \sum_{j \in \hat{\mathcal{Q}}_1} \Phi(y_j, T) + \sum_{j \in \hat{\mathcal{Q}}_2} \Phi(\overline{y}_j, T) \ .$$

where $\hat{\mathcal{Q}}_1 = \{y_j; l_j \neq \alpha\}$ and $\hat{\mathcal{Q}}_1 = \{y_j; l_j = \alpha\}$. Therefore, we will construct the graph using T, partially using indicator y_j, and partially using the negation of indicator $\overline{y}_j$ depending on whether $l_j = \alpha$.

2.5 Learning

The full CRF model in Eq. 2.2 contains several terms. In order to balance the importance of different terms, we introduce a set of linear weights for each term as follows,

$$
\begin{aligned}
W^T\Psi(X.Y) &= \sum_{c\in\mathcal{C}} w_c\psi_c(X_c) + \sum_{(j,k)\in\mathcal{U}_1} w_{l_j,l_k}^{r_{jk}}(y_j,y_k) && (2.16)\\
&+ \sum_{j\in\mathcal{Q}_1} w^u(l_j)(\Phi(y_j,X)+\varphi_u(y_j)) \\
&+ w^{co}(\sum_{(j,k)\in\mathcal{U}_2}\varphi_p(y_j,y_k) + \sum_{j\in\mathcal{Q}_2}\Phi(y_j,X)) \ , && (2.17)
\end{aligned}
$$

where w_c models weights for unary, pair-wise, and higher-order terms in X. $w^u(l)$ is the object category specific weight for unary term in y, w^{co} is the weight for stuff-object-instance indicators, and $w_{l_j,l_k}^{r_{jk}}$ is the pair-wise weights for a specific co-occurrence type r_{jk} related to the pair of object categories l_j, l_k in Eq. 2.6. Recall from Sec. 2.3.2 and 2.4 that $\mathcal{Q}_1$ and $\mathcal{Q}_2$ are the set of indicator variables for things and stuff respectively. Similarly, $\mathcal{U}_1$ and $\mathcal{U}_2$ are the subset of $\mathcal{U}$ such that $j,k\in\mathcal{Q}_1$ or $j,k\in\mathcal{Q}_2$ respectively. Since all these weights are linearly related to the energy function, we formulate the problem of jointly training these weights as a Structured SVM (SSVM) learning problem [121] similar to [28].

Assume that a set of example images, ground truth segment object category labels, and ground truth object bounding boxes $\{I^n, X^n, Y^n\}_{n=1,\ldots,N}$ are given. The SSVM problem is as follows,

$$
\begin{aligned}
&\min_{W,\xi\geq 0} \quad W^TW + C\sum_n \xi^n(X,Y) && (2.18)\\
&\text{s.t.} && (2.19)\\
&\xi^n(X,Y) = \max_{X,Y}(\triangle(X,Y;X^n,Y^n) && (2.20)\\
&\qquad + W^T\Psi(X^n,Y^n) - W^T\Psi(X,Y)), \forall n \ ,
\end{aligned}
$$

where W concatenates all the model parameters which are linearly related to the

potentials $\Psi(X,Y)$; C controls the relative weight of the sum of the violated terms $\{\xi^n(X,Y)\}$ with respect to the regularization term; $\triangle(X,Y;X^n,Y^n)$ is the loss function that generates large loss when the X or Y is very different from X^n or Y^n. Depending on the performance evaluation metric, we design different loss functions as described in the Sec. 2.5.1

Following the SSVM formulation, we propose to use a stochastic subgradient descent method to solve Eq. 2.18. The subgradient of $\partial_W \xi^n(X,Y)$ can be calculated as $\Psi(X^n,Y^n) - \Psi(X^*,Y^*)$, where $(X^*,Y^*) = \arg\min_{X,Y}(W^T\Psi(X,Y) - \triangle(X,Y;X^n,Y^n))$. When the loss function can be decomposed into a sum of local losses on individual segments and individual detections, (X^*,Y^*) can be solved using graph-cut similar to the inference problem (Sec. 2.4). For other complicated loss functions, we found that it is effective to set (X^*,Y^*) approximately as $\arg\min_{X,Y} W^T\Psi(X,Y)$, when the loss is bigger than a threshold.

The remaining model parameters are set as follows. The object category-specific $R(l)$ in Eq. 2.3 are estimated using the median values observed in training data. The γ involved in Eq. 2.8 are estimated from the MSE as described in [80]. The β in Eq. 2.5 are set to be the detection confidence.

2.5.1 Loss Function

For the experiment on Stanford dataset, since the performance is measured by the average classification accuracy across different object categories, we define the following loss function. The overall loss function $\triangle(X,Y;X^n,Y^n)$ is decomposed into sum of the segmentation loss $\triangle(X;X^n)$ and the detection loss $\triangle(Y;Y^n)$.

The segmentation loss $\triangle(X;X^n)$ is defined as

$$\triangle(X;X^n) = \frac{1}{Q}\sum_{i\in\mathcal{V}} \mathbf{1}\{x_i \neq x_i^n\} c_x(l_i) \ , \tag{2.21}$$

where $\mathcal{V}$ captures the indices for the set of segments, $\mathbf{1}\{STATEMENT\}$ is 1 if the $STATEMENT$ is true, $c_x(l_i)$ is the object category l_i dependent cost (used to re-weight the loss contributed from different object categories), and $Q = \sum_{i \in \mathcal{V}} c_x(l_i)$. Therefore, the overall segmentation loss can be decomposed into a sum over local loss for each segment $\frac{1}{Q}\mathbf{1}\{x_i \neq x_i^n\}c_x(l_i)$.

The detection loss $\triangle(Y; Y^n)$ is defined as

$$\triangle(Y; Y^n) = \frac{1}{M} \sum_{i \in \mathcal{B}} \mathbf{1}\{y_i \neq y_i^n\} c_y(l_i) \ , \tag{2.22}$$

where $\mathcal{B}$ captures the indices for the set of detections, $M = \sum_{i \in \mathcal{B}} c_y(l_i)$. Similarly, the overall detection loss can be decomposed into a sum over local loss for each detection $\frac{1}{M}\mathbf{1}\{y_i \neq y_i^n\}c_y(l_i)$.

For the experiment on PASCAL dataset, since the segmentation performance is measured by $\frac{\text{true positive}}{\text{true positive+ false positive + false negative}}$, the overall loss function $\triangle(X, Y; X^n, Y^n)$ is decomposed into sum of the segmentation loss $\triangle(X; X^n)$ and the detection loss $\triangle(Y; Y^n)$. The segmentation loss is 1-segmentation performance and the detection loss is the same as before. Since the segmentation loss cannot be decomposed into a per segment loss, we obtain (X^*, Y^*) approximately as $\arg\min_{X,Y} W^T \Psi(X, Y)$, when $\triangle(X^*, Y^*; X^n, Y^n)$ is bigger than a threshold.

2.6 Experiments

We compare our full ACRF model with [44, 80, 81, 95, 118] on Stanford Background (referred as to Stanford) dataset [44] as well as with several state-of-the-art techniques on PASCAL VOC 2009 (referred as to PASCAL) dataset [31]. As opposed to other datasets, such as MSRC [110], the Stanford dataset contains more cluttered scenes and more "things" object instances per image. Hence, segmenting and detecting "things" is particularly challenging. Conversely, the PASCAL dataset contains

(a) Global Accuracy					
[118]	[44]	[95]	[81]	[80]	ACRF
77.5	76.4	76.9	80.2	80.0	**82.4**

(b)	Back-ground	Car	Person	Motor-bike	Bus	Boat	Cow	Sheep	Bi-cycle	Global	Avg
CRF	77.4	49.1	39.9	15.3	76.3	18.9	65.0	**70.4**	17.3	79.9	47.7
C+D	77.1	56.7	**61.7**	9.3	69.7	**36.9**	88.1	62.8	64.2	82.0	58.5
ACRF	77.2	**74.9**	60.1	**17.2**	**79.4**	**36.9**	**88.6**	58.2	**64.7**	**82.4**	**61.9**

Figure 2.4: Segmentation performance comparison on the Stanford dataset. (a) Global segmentation accuracy of our ACRF model compared with state-of-the-art methods, where "Global" is the overall percentage of pixels correctly classified. (b) System analysis of our model. The CRF row shows the results by using only the stuff-stuff relationship component (first term in Eq. 2.2) of our ACRF model. The C+D row shows results by adding independent detections indicators to the CRF model (first two terms in Eq. 2.2). The last row shows results of the full ACRF model. Notice "Avg." is the average of the percentage over eight foreground classes and one background class.

larger number of "things" labels with a single "stuff" label, with limited number of object instances in each image.

For all the experiments below, we use the same pre-trained LSVM detectors [35] to obtain a set of object-instance hypotheses for "things" categories (e.g., car, person, and bike). The object depths are inferred by combining both cues from the size and the bottom positions of the object bounding boxes similar to [3, 65, 115]. The responses from off-the-shelf stuff classifiers are used as the unary stuff potentials in our model. We model different types of pair-wise stuff relationships using a codebook representation similar to [16].

The following geometric pair-wise thing relationships are used for the experiments to incorporates geometric spatial relationship between two bounding boxes: *next-to*, *above*, *below*, *in-front*, *behind*. On top of that, we have one additional geometric relationship based on horizon lines agreement between two detections.

Geometric spatial relationships are determined by following steps. To establish the geometric relationship given a pair of detection bounding boxes, we firstly set one of them as a reference box. Then, we draw an additional box with respect to a

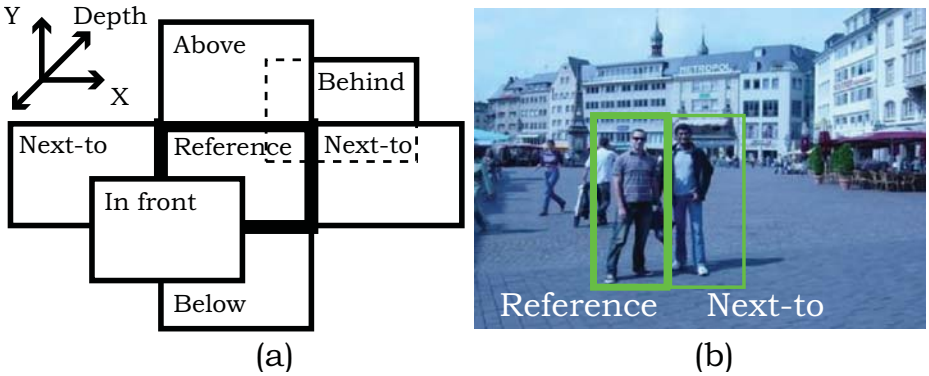


Figure 2.5: (a) Pairwise things relationships can be determined by drawing an additional box with respect to a reference box. (b) In this example, a person (right) is 'next-to' a person on the left side.

reference bounding box for a certain spatial relationship (i.e. *above*: draw on top of a box; *next-to*: draw on left or right side of a box, etc. See Fig. 2.5 for details). If a drawn box overlaps more than 50% with the other detection bounding box which is not selected as a reference box, we can specify a relationship to the given pair of boxes. This procedure is repeated for all geometric relationships.

The additional geometric spatial relationship is based on whether two horizon lines from two bounding boxes are in agreement. In specific, horizon lines for two boxes are estimated assuming objects' average heights, similar to [65]. If two lines are close to each other within a certain range, which is a function of the specific class (i.e. person or car have smaller variance, boat have a larger variance), they are having a same horizon line.

Stanford dataset. The Stanford dataset [44] contains 715 images from challenging urban and rural scenes. On top of 8 background ("stuff") categories, we annotate 9 foreground ("things") object categories - car, person, motorbike, bus, boat, cow, sheep, bicycle, others. We follow the 5-fold cross-validation scheme which splits the data into different 572 training and 143 test images. We use the same STAIR Vision Library [46] used in [44] to obtain the stuff unary potentials. Pixel-wise segmentation performance is shown in Fig. 2.4. Our ACRF model outperforms all state-of-the-

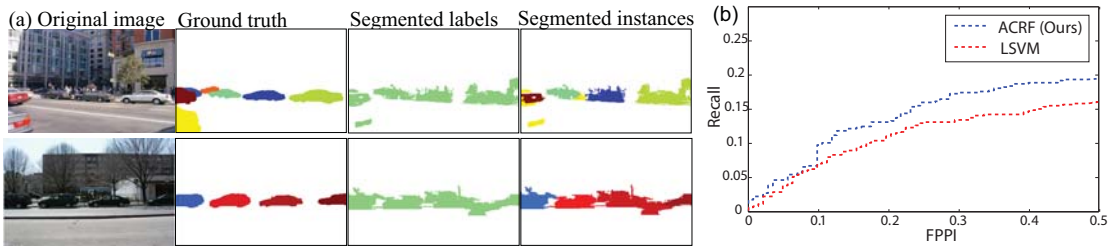


Figure 2.6: (a) Typical thing segmentation results on the Stanford dataset. Notice that our model can obtain instance-based segmentations (last column) due to the ability to reason in the augmented labeling space $\hat{\mathcal{Q}}$. (b) Recall v.s. FPPI curves of our ACRF and LSVM on Stanford dataset. Our ACRF achieves better recall at different FPPI values.

	Background	Aeroplane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dining table	Dog	Horse	Motor bike	Person	Potted plant	Sheep	Sofa	Train	TV/monitor	Average
CRF	69.0	19.7	2.4	8.8	6.8	8.8	21.6	17.5	13.1	0.5	8.6	7.1	6.5	6.5	13.8	19.2	5.8	13.6	3.3	21.3	9.5	13.5
CRF+Det	71.0	21.8	**14.8**	21.1	**18.7**	**34.8**	**48.3**	33.3	16.7	**12.3**	**27.5**	10.5	14.1	27.5	35.4	31.2	29.8	**28.7**	17.5	31.8	27.7	27.3
ACRF	**75.5**	**29.1**	14.7	**23.0**	18.2	34.0	47.8	**40.4**	**17.2**	11.4	27.0	**12.6**	**17.5**	**30.1**	**40.1**	**34.9**	**30.6**	28.2	**20.7**	**31.0**	**30.1**	**29.4**

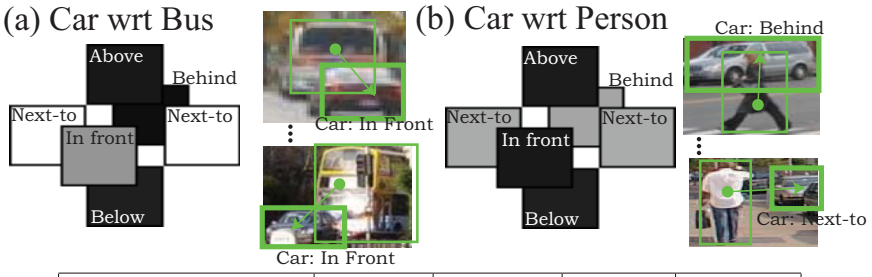
Figure 2.7: The segmentation accuracy of different variants of our model (i.e., CRF, CRF+ Detection, and full ACRF models) on PASCAL dataset.

	Background	Aeroplane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dining table	Dog	Horse	Motor bike	Person	Potted plant	Sheep	Sofa	Train	TV/monitor	Average
BONN	83.9	64.3	21.8	21.7	32.0	40.2	57.3	49.4	38.8	5.2	28.5	22.0	19.6	33.6	45.5	33.6	27.3	40.4	18.1	33.6	46.1	36.3
CVC	80.2	67.1	26.6	30.3	31.6	30.0	44.5	41.6	25.2	5.9	27.8	11.0	23.1	40.5	53.2	32.0	22.2	37.4	23.6	40.3	30.2	34.5
NECUIUC	81.8	41.9	23.1	22.4	22.0	27.8	43.2	51.8	25.9	4.5	18.5	18.0	23.5	26.9	36.6	34.8	8.8	28.3	14.0	35.5	34.7	29.7
ACRF	75.5	29.1	14.7	23.0	18.2	34.0	47.8	40.4	17.2	11.4	27.0	12.6	17.5	30.1	40.1	34.9	30.6	28.2	20.7	31.0	30.1	29.4
UoCTTI	78.9	35.3	22.5	19.1	23.5	36.2	41.2	50.1	11.7	8.9	28.5	1.4	5.9	24.0	35.3	33.4	35.1	27.7	14.2	34.1	41.8	29.0
NECUIUC	81.5	39.3	20.9	22.6	21.7	26.1	37.1	51.5	25.2	5.7	17.5	15.7	24.2	27.4	35.3	33.0	7.9	23.4	12.5	32.1	33.3	28.3
LEAR	79.1	44.6	15.5	20.5	13.3	28.8	29.3	35.8	25.4	4.4	20.3	1.3	16.4	28.2	30.0	24.5	12.2	31.5	18.3	28.8	31.9	25.7
BROOKES	79.6	48.3	6.7	19.1	10.0	16.6	32.7	38.1	25.3	5.5	9.4	25.1	13.3	12.3	35.5	20.7	13.4	17.1	18.4	37.5	36.4	24.8
UCI	80.7	38.3	30.9	3.4	4.4	31.7	45.5	47.3	10.4	4.8	14.3	8.8	6.1	21.5	25.0	38.9	14.8	14.4	3.0	29.1	45.5	24.7
MPI	70.9	16.4	8.7	8.6	8.3	20.8	21.6	14.4	10.5	0.0	14.2	17.2	7.3	9.3	20.3	18.2	6.9	14.1	0.0	13.2	13.2	15.0

Figure 2.8: Segmentation accuracy of our ACRF model compared with other state-of-the-art methods on PASCAL dataset.

art methods [44, 80, 81, 95, 118] [1] (Fig. 2.4(a)). A system analysis of our model (Fig. 2.4(b)) shows that the performances of most foreground classes (seven out of eight) are significantly improved when additional components are added on top of the baseline CRF model, while the performance of the background classes remain almost unchanged. As a result, the full ACRF model obtains the best performance for six out of eight foreground classes and a 14.2% average improvement over the baseline

[1] We implement [80, 81] by ourselves and evaluate the performance.



(c)	Before	BL1	BL2	ACRF
next-to	5.4%	25.9%	32.2%	**92.5%**
above/below	6.1%	51.72%	61.2%	**76.2%**
in-front/behind	5.7%	50.0%	60.4%	**72.3%**
different horizon	0.1%	38.5%	15.0%	**60.0%**

Figure 2.9: Examples of the learned pair-wise things relationships are visualized in panel (a,b). The grayscale color code indicates to what degree the relationship is encouraged (white means it is encouraged, black means it is not encouraged and suppressed). Our model learned that (a) a car is likely to be in front of a bus, and a car is unlikely to be below a bus, (b) a car is likely to be behind a person. (c) Prediction accuracy of the objects co-occurrence for each type of relationship averaged over 5-fold validations. The first and last columns show the accuracy before and after applying inference on our full ACRF model, respectively. Notice that there is a consistent improvement across all types. The performance of two baseline methods are reported in the middle two columns which are all inferior then our results.

model. Typical results are shown in Fig. 2.10-Top. We highlight that our model can generate object instance-based segmentations due to the ability to reason in the *augmented labeling space* $\hat{\mathcal{Q}}$ (Fig. 2.6(a)). Our method can predict the numbers of object instances per image accuractely with an average errors of 0.27.

Another advantage of using our model is to improve detection accuracy. We measured detection performance in terms of Recall v.s. False Positive Per Image (FPPI) in Fig. 2.6(b), where detection results from 5-fold validations are accumulated and shown in one curve. The performance of the proposed model is compared with the pre-trained LSVM [35]. Our model achieves consistent higher recall than the LSVM baseline as shown in Fig. 2.6(b).

PASCAL dataset. This dataset contains 14,743 images with 21 categories including 20 thing categories and 1 stuff category. Only a subset of images have segmentation

labels, and we used the standard split for training (749 images), validation (750 images), and testing (750 images). A system analysis of our model (Fig. 2.7) shows that the performances of most classes were improved when additional components are added on top of the baseline CRF model. Notice that the baseline CRF has a fairly low performance, since we use only the pixel-wise unary responses from the first layer of the hierarchical CRF [80]. However, our ACRF model is able to significantly boost up the performance and achieves competitive accuracy compared other teams in the challenge (ranked in 4_{th} in Fig. 2.8) Moreover, the average in predicting the error of numbers of object instances per image is only 0.06. Typical results are shown in Fig. 2.10-Bottom.

2.6.1 Relationship Analysis

The learned model parameters for a few typical pair-wise things relationships are visualized in Fig. 2.9(a,b). In Fig. 2.9(c), we compare the accuracy of predicting the correct relationship of objects before (i.e., raw detections from LSVM [35]) and after applying inference on our ACRF. A relationship of objects is considered correct if both their object bounding boxes overlap more than 50% with the ground truth bounding boxes. The accuracy reported in Fig. 2.9(c) is the percentage of correct pairs of objects for each type of relationship. Notice that a significant improvement is achieved by our ACRF model over two baseline methods.

Our baseline methods BL1 and BL2 are defined as follows:

BL1 uses only the detection confidence to prune out detections. In specific, for each pair of bounding boxes with a certain relationship, we assign a score as a sum of scores for both bounding boxes from LSVM. Then, we sample $p\%$ of pairs with highest scores, where p is the percentage of correct ratios for a certain relationship from the training set.

BL2 incorporates pairwise object interactions and prune out detections. Again,

for each pair of bounding boxes with a certain relationship, we assign a score as a sum of detection scores for both bounding boxes. Then, we sample pairs within top $p(c_1, c_2)\%$, where $p(c_1, c_2)$ is a class-pair specific percentage of correct ratios from the training set, and c_1 and c_2 is classes corresponding to two bounding boxes.

Using the inferred relationships we can provide high level geometrical description of the scene and determine properties such as: object x is behind object y. Finally, we can obtain 3D pop-up models of the scene from a single image as in Fig. 2.11

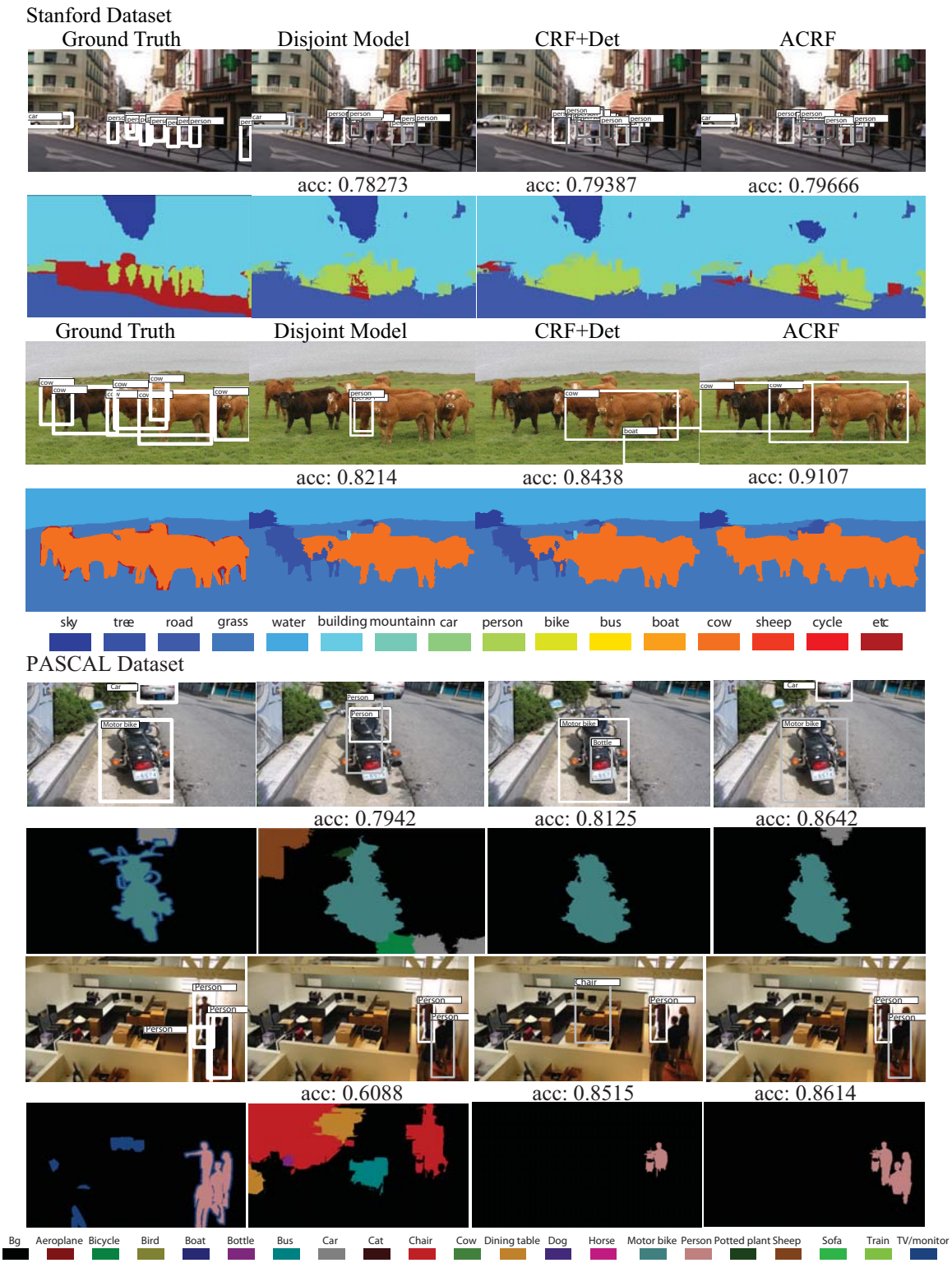


Figure 2.10: Typical results on Stanford (top 4 rows) and PASCAL datasets (bottom 4 rows). Every set of results compare ground truth annotation, disjointed model (disjointedly applied object detection and segmentation), CRF+Det, ACRF, from left to right, respectively. The odd rows show the top K object hypotheses (color-coded bounding boxes representing the confidence ranking from light to dark), where K is the number of recalled objects in the ACRF result. The even rows show the segmentation results (color-code is shown at the bottom).

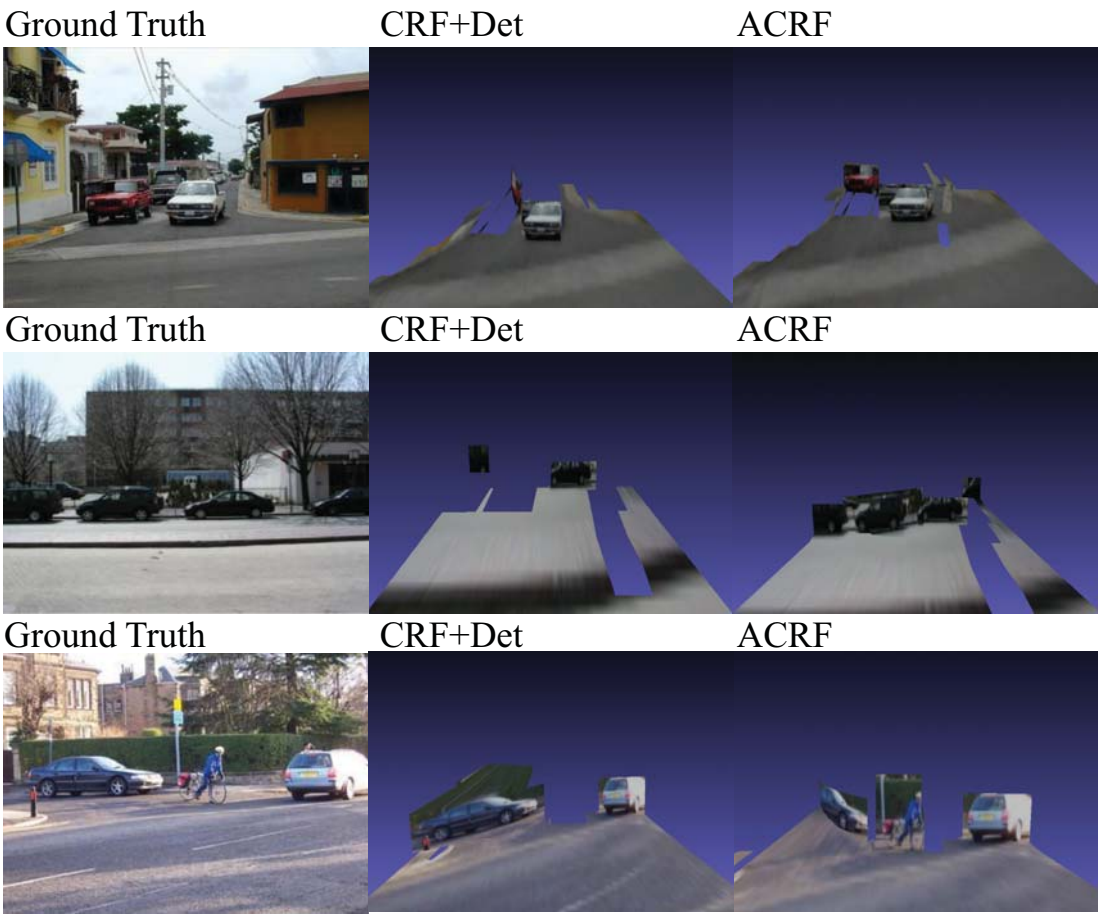


Figure 2.11: 3D pop-up models from Stanford dataset. Videos related to above 3D pop-up models can be found in the project page.

2.7 Conclusion

We have presented a unified CRF-based framework for jointly detecting and segmenting "things" and "stuff" categories in natural images. We have shown that our framework incorporates in a coherent fashion various types of (geometrical and semantic) contextual relationships by using property list. Our new formulation generalizes previous results based on CRF where the focus was to capture the co-occurrence between stuff categories only. We have quantitatively and qualitatively demonstrated that our method: i) produces better segmentation results than state-of-the art on the

Stanford dataset and competitive results on PASCAL'09 dataset; ii) improves the recall of object instances on Stanford dataset; iii) enables the estimation of contextual relationship among things and stuff. Extensions for future work include incorporating more sophisticated types of properties.

CHAPTER III

3D Scene Understanding by Voxel-CRF

In the past few years, researchers have taken advantage of the recent diffusion of depth-RGB (RGB-D) cameras to help simplify the problem of inferring scene semantics. However, while the added 3D geometry is certainly useful to segment out objects with different depth values, it also adds complications in that the 3D geometry is often incorrect because of noisy depth measurements and the actual 3D extent of the objects is usually unknown because of occlusions. In this work we propose a new method that allows us to jointly refine the 3D reconstruction of the scene (raw depth values) while accurately segmenting out the objects or scene elements from the 3D reconstruction. This is achieved by introducing a new model which we called Voxel-CRF. The Voxel-CRF model is based on the idea of constructing a conditional random field over a 3D volume of interest which captures the semantic and 3D geometric relationships among different elements (voxels) of the scene. Such model allows to jointly estimate (1) a dense voxel-based 3D reconstruction and (2) the semantic labels associated with each voxel even in presence of partial occlusions using an approximate yet efficient inference strategy. We evaluated our method on the challenging NYU Depth dataset (Version 1 and 2). Experimental results show that our method achieves competitive accuracy in inferring scene semantics and visually appealing results in improving the quality of the 3D reconstruction. We also demonstrate an interesting application of

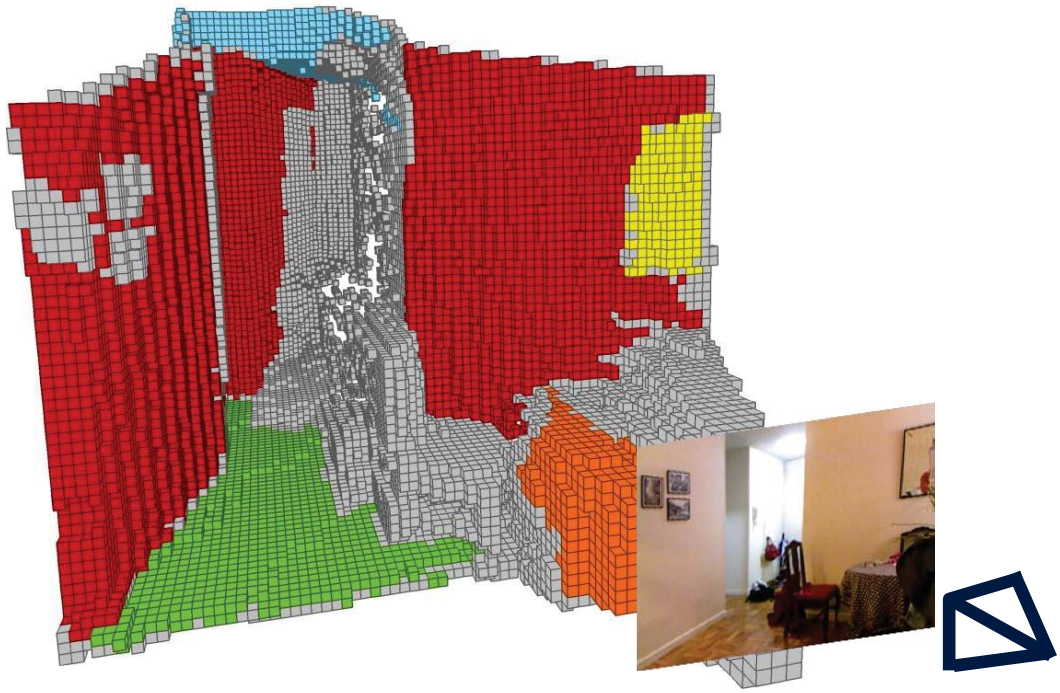


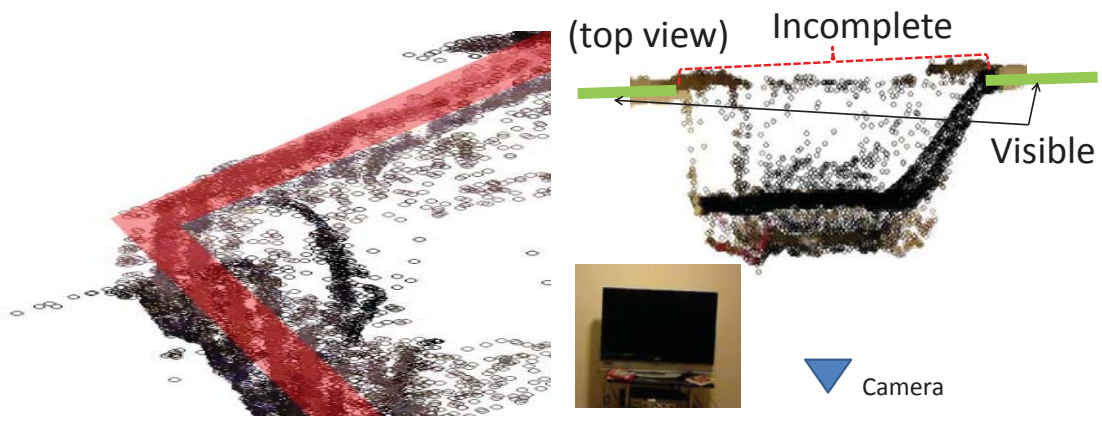
Figure 3.1: Given a single depth-RGB image, our proposed Voxel-CRF (V-CRF) model jointly estimates (1) a dense voxel-based 3D reconstruction of the scene and (2) the semantic labels associated with each voxel. In the figure, red corresponds to 'wall', green to 'floor', orange to 'table' and yellow to 'picture'.

object removal and scene completion from RGB-D images.

3.1 Statement

Understanding the geometric and semantic structure of a scene (scene understanding) is a critical problem in various research fields including computer vision, robotics, and augmented reality. For instance, consider a robot in the indoor scene shown in the Fig. 3.1. In order to safely navigate through the environment, the robot must perceive the free space of the scene accurately (geometric structure). Moreover, in order for the robot to effectively interact with the environment (*e.g.,* to place a bottle on a table), it must recognize the objects in the scene (semantic structure).

Several methods have been proposed to solve the problem of scene understanding



(a) (b)

Figure 3.2: (a) Reconstructed point cloud taken from the corner of the room. Ground truth '*wall*' is highlighted with a red mask. Reconstructing reliable 3D geometry from noisy point cloud is a challenging task. (b) Point clouds do not completely describe the 3D scene. For example, the wall behind the *tv* cannot be reconstructed from depth map.

using a single RGB (2D) image. For instance, in [70, 72, 79, 110], the problem is formulated in terms of the estimation of a consistent set of semantic labels of local image regions (patches or pixels) assuming a *flat* image world. Although the results were promising, such methods do not provide information about the geometric structure of the scene. Recently, attempts have been made to jointly estimate the 3D and semantic properties of a scene using a single image or multiple images [2, 22, 53, 61, 64]. The efficacy of such methods in perceiving the scene geometry, however, is limited due to the inherent geometric ambiguity in a single image. To overcome the ambiguity, researchers have considered using depth and RGB image data for scene understanding [75, 96, 112]. Instead of labeling local 2D image regions, these methods provide semantic description of 3D elements (point clouds) acquired by a RGB-D camera [94]. However, they rely on the assumption that the 3D structure from the RGB-D device is accurate. This is not always the case due to photometric interference, discretization error, etc (see Fig. 3.2 for typical noisy reconstruction).

In this work, we propose a method to jointly estimate the semantic and geometric

structure of a scene given a single RGB-D image. Unlike [75, 96, 112] where the true geometric structure is assumed to be given and fixed, we represent a scene with a set of small cubic volume (*voxel*) in the space of interest. We jointly estimate both the semantic labeling and 3D geometry of voxels of the scene given a noisy set of inputs. This allows us to *i)* correct noisy geometric estimation in input data and *ii)* provide the interpretation of non-visible geometric elements (such as the wall occluded by the table in Fig. 3.1). Our method is based on a voxel conditional random field model which we have called Voxel-CRF (V-CRF). In our V-CRF model, each node represents a voxel in the space of interest. A voxel may or may not include one or multiple points acquired by the RGB-D sensor. The state of each voxel is summarized by two variables, *occupancy* and *semantic label*. An auxiliary variable *visibility* is introduced to help relate voxels and 2D RGB or depth observation (Sec. 3.3). Semantic and geometric interpretation of a scene is achieved by finding the configuration of all variables that best explains the given observation.

The configuration of variables in the V-CRF model needs to be consistent with certain important geometric and semantic rules that ensure stable and more accurate 3D reconstruction and classification of the elements in the scene. This includes relationships such as '*supported by*' or '*attached to*' (Sec. 3.4.2). Geometric and semantic relationships based on higher-level elements such as certain groups of voxels which belong to the same plane (or object) are encoded using interactions between groups of voxels. These relationships are especially useful for consistent labeling of voxels in an occluded space (Sec. 3.4.3). The parameters associated with the above-mentioned interaction functions are learned from training data.

Given our V-CRF model, we solve the scene understanding problem by minimizing the energy function associated with the V-CRF. Instead of assuming that the true 3D geometry is given, we jointly estimate the geometric and semantic structure of the scene by finding the best configuration of all occupancy and semantic label variables of

all voxels in the space. Our inference algorithm iterates between 1) deciding voxels to be associated with observations and 2) reasoning about the geometric and semantic description of voxels. In each iteration, we obtain an approximate solution using graph-cuts based inference [17].

In summary, the contributions of this work are 5 folds. 1) We propose a new voxel based model for 3D scene understanding with RGB-D data that jointly infers the geometric and semantic structure of the scene (Sec. 3.3). 2) We improve structure estimation given noisy and incomplete 3D reconstruction provided by RGB-D sensors. 3) Geometric and semantic rules are proposed and modeled in the V-CRF model (Sec. 3.3&3.4). 4) An efficient iterative method is proposed for performing inference in the V-CRF model (Sec. 3.5). 5) We demonstrate (through qualitative and quantitative results and comparisons on benchmarks) that V-CRF produces accurate geometric and semantic scene understanding results (Sec. 3.6). Some applications enabled by the V-CRF model are discussed in Sec. 3.6.4.

3.2 Related Work

Our model is related to [53, 68] in that blocks are used to represent 3D space. On the other hand, unlike [53, 68], our blocks are defined at a fine resolution that enables us to understand scenes (such as cluttered indoor environments) in more detail. The methods proposed in [55, 61, 107] are also relevant to our work. These methods analyze geometric properties of the underlying scene and infer free space. However, our model can produce more fine grained labeling of geometric and semantic structure which is important for cluttered scenes. The approaches for scene understanding described in [56, 75, 96, 112] are based on RGB-D data. Similar to our method, these methods assign semantic labels to image elements such as 3D points or super-pixels. However, image elements in these works are defined only over visible elements. Also, it is assumed that image elements are already correctly localized in 3D space. In

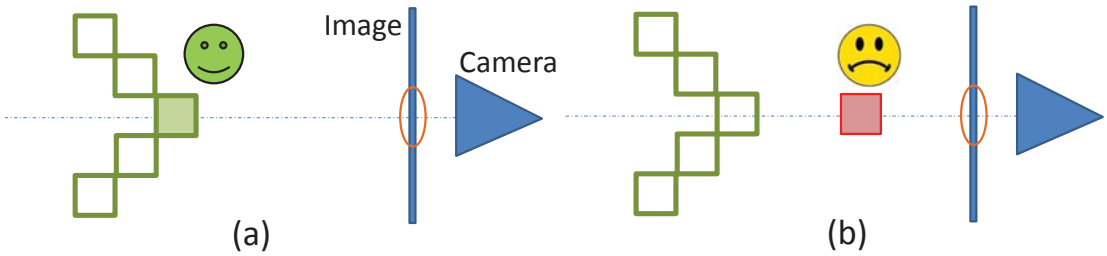


Figure 3.3: Ambiguity of assigning image observations to the voxels in a view ray. Five voxels with green outline are the ground truth voxels in a correct place. (a) For the successful cases, the voxel can be reconstructed from a depth data. (b) Unfortunately, due to noisy depth data, incorrect voxels are reconstructed in many cases.

contrast, our model can reason about the labeling of both visible and occluded image elements.

Our work is also closely related to [8, 82] in the use of a random field model for joint semantic labeling and geometric interpretation. [82] encouraged consistent semantic and geometric labeling of pixels by penalizing sudden changes in depth or semantic labeling results. [8] showed that the joint geometric-semantic labeling model helps in geometry estimation. Similar to our occlusion reasoning, they showed that the depth of fully occluded regions can be inferred by having stereo images. However, they did not consider a complete reconstruction of the scene. The problem of labeling occluded regions is also discussed in [51], where relative relationships between objects and background are used to infer labels of the occluded region. However, the lack of a voxel representation restricts [51] to reconstruction of the foreground and background layers. In contrast, in theory, our model can reconstruct any number of layers in the scene.

3.3 Voxel-CRF

We now describe the Voxel-CRF (V-CRF) model. We represent the semantic and geometric structure of the scene with a 3D lattice where each cell of the lattice is a

voxel. V-CRF is defined over a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{C})$, where $\mathcal{V}$ are vertices representing voxels, edges $\mathcal{E}$ connect horizontally or vertically adjacent pairs of vertices, cliques $\mathcal{C}$ are groups of voxels which are related, *e.g.,* voxels on the same 3D plane, or voxels that are believed to belong to an object (through an object detection bounding box). The state of each voxel is described with a structured label $\ell_i = (o_i, s_i)$ and the visibility v_i. The first variable o_i represents voxel occupancy; *i.e.,* it indicates whether voxel i is *empty* ($o_i = 0$) or *occupied* ($o_i = 1$). The second variable s_i indicates the index of semantic class the voxel belong to; *i.e.,* $s_i \in \{1, \cdots, |S|\}$ if the voxel is occupied ($o_i = 1$), or $s_i = \emptyset$ if $o_i = 0$, where $|S|$ is the number of semantic classes (*e.g.,* table, wall, ...). Estimation of the structured label $L = \{\ell_i\}$ over the V-CRF model produces a geometric and semantic interpretation of the scene.

The variable v_i encodes the visibility of a voxel i where $v_i = 1$ and $v_i = 0$ indicate whether the voxel is *visible* or *non-visible*, respectively. Any given ray from the camera touches a single visible (occupied) voxel. Due to the high amount of noise in the RGB-D sensor, it is difficult to unambiguously assign 2D observations (image appearance and texture) to voxels in 3D space (see Fig. 3.3 for an illustration). The visibility variables v_i allow us to reason about this ambiguity. Provided that we know which single voxel is visible on the viewing-ray, we can assign the 2D image observation to the corresponding voxel. Since *visibility* is a function of *occupancy*, and vice versa, we infer the optimal configuration of the two in an iterative procedure.

V-CRF can be considered as a generalization of existing CRF-based models for scene understanding in 3D [75, 96, 112], where $\{o_i\}$ and $\{v_i\}$ are assumed to be given, and semantic labels $\{s_i\}$ are inferred only for visible and occupied scene elements. In contrast, V-CRF model is more flexible by having o_i and v_i as random variables, and this enables richer scene interpretation by *i)* estimating occluded regions, *e.g.,* (o_i, s_i)=(*occupied*,*table*), v_i= *occluded*, and *ii)* correcting noisy depth data.

3.4 Energy Function

Given a graph $\mathcal{G}$, we aim to find $V^* = \{v_i^*\}$ and $L^* = \{\ell_i^*\}$ that minimize the energy function $E(V, L, O)$, where $O = \{C, I, D\}$, C is the known camera parameters, I is the observation from a RGB image, and D is the observation from a depth map. The energy function can be written as a sum of potential functions defined over individual, pairs, and group of voxels as: $E(V, L, O) =$

$$\sum_i \phi_u(v_i, \ell_i, O) + \sum_{i,j} \phi_p(v_i, \ell_i, v_j, \ell_j, O) + \sum_c \phi_c(V_c, L_c, O) \tag{3.1}$$

where i and j are indices of voxels and c is the index of higher-order cliques in a graph. The first term models the observation cost for individual voxels, while the second and third terms model semantic and geometric consistency among pairs and groups of voxels, respectively.

3.4.1 Observation for Individual Voxels

The term ϕ_u represents the cost of the assignment (v_i, ℓ_i) for a voxel i. We model the term for two different cases, when voxel i is occupied $(o_i = 1)$ and when it is empty $(o_i = 0)$.

$$\phi_u(v_i, \ell_i, O) = \begin{cases} k_1 & \text{if } o_i = 1, s_i \neq \emptyset \\ k_2 & \text{if } o_i = 0, s_i = \emptyset \end{cases} \tag{3.2}$$

where k_1 and k_2 are defined as $k_1 =:$

$$w_1^u v_i \log P(s_i|O) - w_2^u \log f_s(d_i - d_{r(i)}^m) - w_3^u \log \frac{|\mathrm{P}_i|}{|\mathrm{P}_i^{max}|} \tag{3.3}$$

$$\text{and } k_2 = -w_4^u \log(1 - f_s(d_i - d_{r(i)}^m)) - w_5^u \log(1 - \frac{|\mathrm{P}_i|}{|\mathrm{P}_i^{max}|}). \tag{3.4}$$

When the voxel i is occupied $(o_i = 1)$, it is composed of three terms. The first term incorporates the observations $P(s_i|O)$ from an image if it is visible $(v_i = 1)$, to

estimate a structured label of the voxel. The second term models the uncertainty in the depth value from the RGBD image through a normal distribution $f_s \sim \mathcal{N}(0, \sigma^2)$. Larger the disparity between depth according to the data map value $d_r^m(i)$, which is value associated with a ray $r(i)$ for a voxel i, and the voxel i's depth d_i, more likely it is to be labeled as an empty state. The third term models the occupancy based on density of 3D points in a voxel i. Note that there can be more than one image pixel corresponding to a voxel. We measure the ratio $|\mathrm{P}_i|/|\mathrm{P}_i^{max}|$, which is the ratio between the number of detected points in 3D cubical volume associated with a voxel i over the maximum number of 3D points in a voxel i, *i.e.,* the number of rays penetrating through a voxel i. If there is an object at voxel i, and the surface is perpendicular to the camera ray, the number of points is the largest. If this ratio is small (*i.e.* few points), the energy function encourages $o_i = 0$.

In the case the voxel i is empty ($o_i = 0$), the energy models the sensitivity of the sensor (first term) and the density of point clouds (second term). Different terms are balanced with weights $w_{\{\cdot\}}^u$, which are learned from the training dataset as discussed in Sec. 3.5.

3.4.2 Relating Pairs of Voxels

The pairwise energy terms penalize labeling results of pairs of voxels that are geometrically or semantically inconsistent. Two different types of neighborhoods are considered to define pairwise relationships between voxels: *i)* adjacent voxels in 3D lattice structure, and *ii)* adjacent voxels in its 2D projection. The pairwise costs depend on visibility, spatial relationship, and appearance similarity of a pair of voxels. Appearance similarity between a pair of voxels (*e.g.,* color) is represented by c_{ij} which is a discretized color difference between voxels i and j, similar to [43]. If voxel j is empty or occluded, we use c_i, *i.e.* in this case the cost is the function of the color of the visible voxel i. The pairwise cost on the labeling of voxels also depends on their

visibility and is defined as:

$$\phi_p(v_i = 1, \ell_i, v_j = 1, \ell_j) = w_1^{pw}(s_{ij}, c_{ij})T[\ell_i \neq \ell_j] \tag{3.5}$$

$$\phi_p(v_i = 0, \ell_i, v_j = 1, \ell_j) = w_2^{pw}(s_{ij}, c_j)T[\ell_i \neq \ell_j] \tag{3.6}$$

$$\phi_p(v_i = 1, \ell_i, v_j = 0, \ell_j) = w_3^{pw}(s_{ij}, c_i)T[\ell_i \neq \ell_j] \tag{3.7}$$

$$\phi_p(v_i = 0, \ell_i, v_j = 0, \ell_j) = w_4^{pw}, \tag{3.8}$$

where $T[\cdot]$ is the indicator function, s_{ij} is a spatial relationship between voxels i and j. i and j are chosen differently for 2D and 3D cases as discussed below. These functions penalize if ℓ_i and ℓ_j are inconsistent. The exact penalty for inconsistent assignments depends on the relative spatial location s_{ij} and colors c_{ij} of the voxel pairs. $w_{\{\cdot\}}^{pw}(s_{ij}, c_{ij})$ are weights that are learned from the training data.

Adjacent pairs in 3D. For all adjacent pairs of voxels, we specify their spatial relationship s_{ij}, where $s_{ij} \in \{vertical, horizontal\}$. The color difference between i and j is also used to modulate the cost $w_{\{\cdot\}}^{pw}(\cdot)$, where we cluster color difference between two voxels as in [43], c_{ij} is the index of a closest cluster.

Adjacent pairs in 2D. On top of adjacent voxels in 3D, the adjacency between a pair of voxels in the projected 2D images is formulated as pairwise costs. For example, occlusion boundaries are useful cues to distinguish voxels that belong to different objects; if two voxels are across a detected occlusion boundary (when projected in the view of the camera), they are likely to have different semantic labels. On the other hand, if two voxels across the boundary are still close in 3D, they are likely to have a same semantic label. The relationship of voxels are automatically indexed as follows. First, we extract pairs of 2D pixels from 2D RGB images which are on the opposite side of the occlusion boundaries. The pair of 2D pixels are then projected into 3D voxels. From the training data, we collect the relative surface feature between

voxels i and j[1] and cluster them to represent different types of corners, depending on their geometric properties in 3D. Finally, the spatial index s_{ij} indicates a cluster ID. We learn different weights for different cluster automatically from the training data.

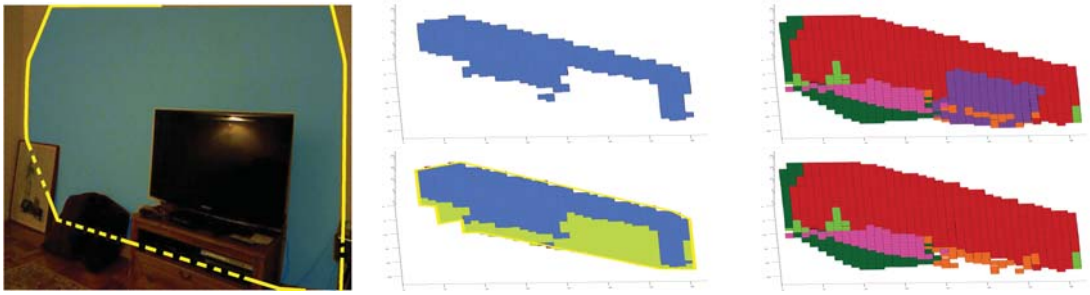
3.4.3 Relating Groups of Voxels

We now introduce higher-order potentials that encode the relationship among more than two voxels. The potentials enforce semantic and geometric consistency among voxels in a clique $c \in \mathcal{V}_C$ of voxels that can be quite far from each other. The relationships for a group of voxels can be represented using the Robust Pott's model [72]. Different types of 3D priors can be used, *e.g.,* surface detection, object detection, or room layout estimation; however, in this work, we consider two types of voxel groups $\mathcal{V}_C$, 1) 3D surfaces that are detected using a Hough voting based method [13] and 2) categorical object detections [36] as follows[2].

3D Surfaces. The first type is the group of voxels that belong to a 3D surface (wall, tables etc). From the depth data and its projected point clouds, we can identify 3D surfaces [13] and these are useful to understand images for two reasons. First, a surface is likely to belong to an object or a facet of the indoor room layout, and there is consistency among labels of voxels for a detected plane. Second, the part of the plane occluded by other objects can be inferred by extending the plane to include the convex hull[3] of the detected surface (See Fig. 3.4). According to the law of closure of Gestalt theory, both visible and invisible regions inside this convex hull are likely to belong to the same object.

Object Detections. Object detection methods provide a cue to define groups of voxels (bounding box) that take the same label, as used for 2D scene understanding in [70, 81], where we grouped a set of visible voxels which fall inside in the object

[1]The surface feature for adjacent regions i and j is composed of surface norm, color, and height.
[2]Room layout estimation is not used due to heavy clutter in the evaluated dataset.
[3]3D plane with smallest perimeter containing all the points associated with a detected surface.



(a) (b) (c)

Figure 3.4: (Best visible in a high resolution) (a) A detected plane using [13] is highlighted with the blue mask. Its convex hull is drawn with the yellow polygon and it includes both visible and occluded region of a planar surface. (b) A group of voxels associated with the detected planar surface (top) and a group of voxels associated with the convex hull (bottom). The voxels in the convex hull not only enforce consistency for visible voxels, but also for occluded voxels. (c) V-CRF result: our model not only allows the labeling of visible voxels for TV (top), but also the labeling of the occluded region corresponding to the 'wall'. For visibility, we removed the voxels corresponding to the TV. (bottom).

bounding box. We use off-the-shelf detectors, *e.g.,* proposed in [36], to find 2D object bounding boxes and then find the corresponding voxels in 3D to form a clique.

3.4.4 Relating Voxels in a Camera Ray

V-CRF model enforces that there is only one visible voxel for each ray from a camera. This is enforced by the following energy term.

$$\phi_c(V_{c_r}, L_{c_r}, O) = \begin{cases} 0 & \text{if } \sum_{i \in c} v_i = 1 \\ \infty & \text{otherwise} \end{cases} \tag{3.9}$$

where c_r is indices of voxels in a single ray.

3.5 Inference and Learning

In this section, we discuss our inference and learning procedures. We propose an inference method where structured labels $\{\ell_i\}$ and visibility labels $\{v_i\}$ are iteratively

updated (Sec. 3.5.1). The parameters of the model are learned using Structural SVM framework [69] (Sec. 3.5.2).

3.5.1 Inference

We find the most probable labelling of L and V under the model by minimizing the energy function Eq. 3.1. Efficiency of the inference step is a key requirement for us as V-CRF is defined over a voxel space which can be much larger than the number of pixels in the image. We propose an efficient graph-cut based approximate iterative inference procedure that is described below.

In the t^{th} iteration, we estimate the value of the visibility variables V_t from L_{t-1} by finding out the first occupied voxel in each ray from a camera. Given V_t, we solve the energy minimization problem $\text{argmin}_L E(V_t, L, O)$ instead of Eq. 3.1, and update L_t. This procedure is illustrated in Alg. 1. Note that, by fixing V_t, the energy (Eq. 3.1) becomes independent of V and can be minimized using graph-cut [17, 72].

Algorithm 1: Iterative inference process for L and V.

0. Initialize V_t, $t = 0$.;
1. Build a V-CRF with unary, pairwise and higher-order potential terms, by fixing V_t.;
2. (Scene understanding) Solve $L_{t+1} = \text{argmin}_L E(V_t, L, O)$ with the graph-cut method;
3. (Updating visibility) From L_{t+1}, update V_{t+1};
4. Go back to Step. 1.;

3.5.2 Learning

The energy function introduced in Sec. 3.4 is the sum of unary, pairwise, and higher-order potentials. Since the weights $W = (w^u_{\{.\}}, w^{pw}_{\{.\}}, w^g_{\{.\}})$ are linear in the energy function, we formulate the training problem as a structural SVM problem [69].

Specifically, given N RGB-D images $(I^n, D^n)_{n\in 1\sim N}$ and their corresponding ground truth labels L^n, we solve the following optimization problem:

$$\min_{W,\xi\geq 0} W^T W + C\sum_n \xi^n(L) \tag{3.10}$$
$$s.t. \quad \xi^n(L) = max_L(\Delta(L; L^n) + E(L^n|W) - E(L|W))$$

where C controls the relative weights of the sum of the violated terms $\{\xi^n(L)\}$ with respect to the regularization term. $\Delta(L; L^n)$ is the loss function for the visible voxels according to its structured label that guarantees larger loss when L is more different from L^n. Note that the loss function can be decomposed into a sum of local losses on individual voxels, and the violated terms can be efficiently inferred by the graph-cut method. Similar to [70], stochastic subgradient decent method is be used to solve Eq. 3.10.

3.6 Experiments

We evaluate our framework on two datasets [111, 112].

3.6.1 Implementation Details

Appearance Term. For the appearance term $P(s_i|O)$ for visible voxels in Sec. 3.4.1, we incorporate responses of [79] and [103], which are state-of-the-art methods using 2D and 3D features, respectively.

3D Surface Detection. We find groups of voxels composing 3D surfaces using off-the-shelf plane detector [13], which detects a number of planes from point clouds by hough voting in a parameterized space. Different types of parameterized space can be used; in this work, we used Randomized Hough Voting. Please see [13] for details.

Object Detection. We use pre-trained DPM detector [36] Release 4 [38] to provide detections for higher-order cliques. Among various semantic classes, we used

	[79]	[103]	**U**	**U+PW**	**U+PW+G**
Geo	76.6	80.0	85.8	87.4	**87.7**
S,1^{st}	19.1	38.3	40.4	41.1	41.6
S,5^{th}	-	-	41.7	43.7	**44.6**

Table 3.1: Top-view analysis for NYUD-V1. Different columns are for benchmark methods [79, 103] and different components of our model (U:only unary terms, $U + PW$:unary and pairwise, and $U + PW + G$:full model). Geometric accuracies are reported in the first line. Semantic accuracies (2nd and 3rd lines) is measured after 1st and 5th iterations of inference steps. By having more components, our model gradually improves the accuracy, and iterative procedure further helps. Full model V-CRF achieves the state-of-the-art performance of 87.7% and 44.6% for geometric and semantic estimation accuracy, respectively. The typical examples can be found from Fig. 3.5.

reliable detection results from sofa, chair, and tv/monitors.

Voxel Initialization. To build V-CRF model, the 3D space of interest is divided with voxels having size of $(4cm)^3$ for testing. For training, voxels are divided into $(8cm)^3$ for efficiency. Since the difference in resolution is small we could use the relationships learned from the training set on the test set with reasonable results. Initialization is performed by assigning appearance likelihood for each point in a cloud to a voxel. Note that more than one point from a cloud can be associated with a single voxel; for simplicity, we used averaged appearance likelihood responses from multiple points for Eq. 3.2.

3.6.2 NYU DEPTH Ver. 1

We first evaluate our framework on the NYU Depth dataset Ver. 1 (NYUD-V1) [111], where pixelwise annotations are available for 13 classes. The dataset contains 2347 images from 64 different indoor environments. We used the same 10 random splits of training and testing set used in [103] and compared the performance against [79, 103] as well as variants of our model.

The proposed framework solves semantic and geometric scene understanding jointly. Yet, evaluating the accuracy in 3D is not an easy task because of the lack of ground

truth geometry due to the noisy depth data and incomplete region of occluded part. We propose two metrics for evaluating accuracy - one based on a top view analysis and one evaluating only the visible voxels.

Metric 1: Top-view analysis. Similar to [61, 107], top-view analysis can help understand the results of the framework and perceive the free space of the scene as well as the occluded regions. While [111] only provides frontal view annotation, we annotated top-view ground truth labels as depicted in Fig. 3.5 (d), where free space and object occupancy as well as semantic labeling can be evaluated. We propose a novel user interface for efficient top-view annotation [101]. Specifically, 1320 images from 54 different scenes are annotated[4], where the labeling space is {*empty*, *bed*, *blind*, *window*, *cabinet*, *picture*, *sofa*, *table*, *television*, *wall*, *bookshelf*, *other objects*}.

Fig. 3.5 shows typical examples of scene understanding from single view RGB-D images from the proposed V-CRF. Note that our model improves reconstruction errors in depth map as well as semantic understanding against a benchmark method, *e.g.,* [103]. Fig. 3.6 illustrates the results for different number of iterations; we observe that most of minor errors are corrected in the first iteration, whereas more severe errors are gradually improved over the iterative inference process.

Quantitative results can be found in Table. 3.1. The free space estimation accuracy is measured by evaluating binary classification results for occupancy (empty/non-empty) from the top-view of the image (Table. 3.1, 1st line 'Geo'). The occupancy map from the top-view is an important measure and relevant to a number of applications such as robotics. Compared to [103], our method achieves 7.7% overall improvement. Especially, our unary potential gives 5.8% boost over [103] (pairwise potentials and higher-order potentials further improves the accuracy). Note that our unary potential not only models appearance but also models geometric properties of the occupancy. This allows V-CRF model to achieve better performance even with

[4]Bedroom, kitchen, livingroom, office scenes are annotated.

	[79]	[103]	**U**	**U+PW**	**U+PW+G**
S,5^{th}	42.8	65.5[5]	69.5	69.9	**70.0**

Table 3.2: Visible voxel analysis for NYUD-V1. Semantic labeling accuracies of the visible voxel, after 5th iteration of the inference. Full V-CRF (U+PW+G) model achives the best performance compared against [79, 103] and variants of our models (U, U+PW).

the simple unary model, compared to [103].

We also observe that semantic labeling accuracy is simultaneously improved in Table. 3.1, the second and the third lines. Here, we analyze *i)* the effect of different energy terms and *ii)* the effect of the iterative procedure. It shows that our full model with larger number of iterations achieves the state-of-the-art average accuracy of 44.6%, which is 6.3% higher than the projected results from [103]. The typical examples can be found in Fig. 3.5.

Metric 2: Visible voxels. The accuracy of semantic labels for visible voxels is presented in Table. 3.2. For this evaluation, we used the original labeling over 2347 images with 13 classes annotations [111]. Compared to the state-of-the-art method [103], our full model achieves 4.5% improvement in average recall rate.

3.6.3 NYU DEPTH Ver. 2

The NYU Depth dataset Ver. 2 (NYUD-V2) [112] contains 1449 RGB-D images collected from 464 different indoor scenes having more diversity than NYUD-V1. We split the data into 10 random sets for training and testing and evaluate performance for top-view labeling, and for visible voxels, as in NYUD-V1. The experimental results show that the accuracy is worse than that of NYUD-V1 due to diversity of the dataset, but still full V-CRF model achieves the best performance compared against [79, 103].

[5]This number is equivalent to 2D semantic labeling accuracy 76.1% reported in super-pixel-based evaluation [103]. 2D super-pixel-based evaluation cannot address the accuracy of 3D scene labeling and tends to penalize less for inaccurate labeling for distant 3D regions.

	[79]	[103]	**U**	**U+PW**	**U+PW+G**
Geo (top)	73.2	78.2	85.0	**87.1**	**87.1**
S,5^{th} (top)	16.3	23.9	31.0	32.9	**33.6**
S,5^{th} (visible)	38.6	53.7	61.3	63.2	**63.4**

Table 3.3: The evaluation results on NYUD-V2. The first two lines are for top-view analysis, and the third line is the analysis for visible voxels. The accuracy is worse than that of NYUD-V1 due to diversity in the dataset. Still, our methods achieves the highest accuracy for both geometry estimation and semantic labeling tasks.

Metric 1: Top-view analysis. We annotated top-view with the same labeling space used for NYUD-V1. This consist of 762 images from 320 different indoor scenes. The first and the second rows in Table. 3.3 show the performance of geometry estimation and semantic labeling from the top view, respectively. Our model achieves the best performance in both semantic and geometric accuracy (9.7% and 8.9% improvement over [103]).

Metric 2: Visible voxel. The third row in Table. 3.3 shows semantic labeling accuracy for visible voxels. Our full model achieves 63.4% (9.7% improvement over [103]).

3.6.4 Augmented Reality: Object Removal.

One interesting application is an augmented reality scenario where one can remove or move around objects. This is not possible in most of conventional augmented reality methods [122] where one can put a new object in a scene but cannot remove the existing objects, since it requires a model to *i)* identify semantic and geometric properties of the objects, *ii)* estimate occluded region behind the object. In contrast, V-CRF model can solve this problem. Fig. 3.7 shows that our model can be used to detect, say, a TV set in the scene and remove it. Note that the occluded region behind the TV is reconstructed using pairwise relationships among voxels as discussed in Sec. 3.4.2 and the concept of 3D surface prior as introduced in Sec. 3.4.3.

3.7 Conclusion

We have presented the V-CRF model for jointly solving the problem of semantic scene understanding and geometry estimation that incorporates 3D geometric and semantic relationships between scene elements in a coherent fashion. Our formulation generalizes many existing 3D scene understanding frameworks. Experimental results indicate that our method quantitatively and qualitatively achieves good performance on the challenging NYU Depth dataset (Version 1 and 2).

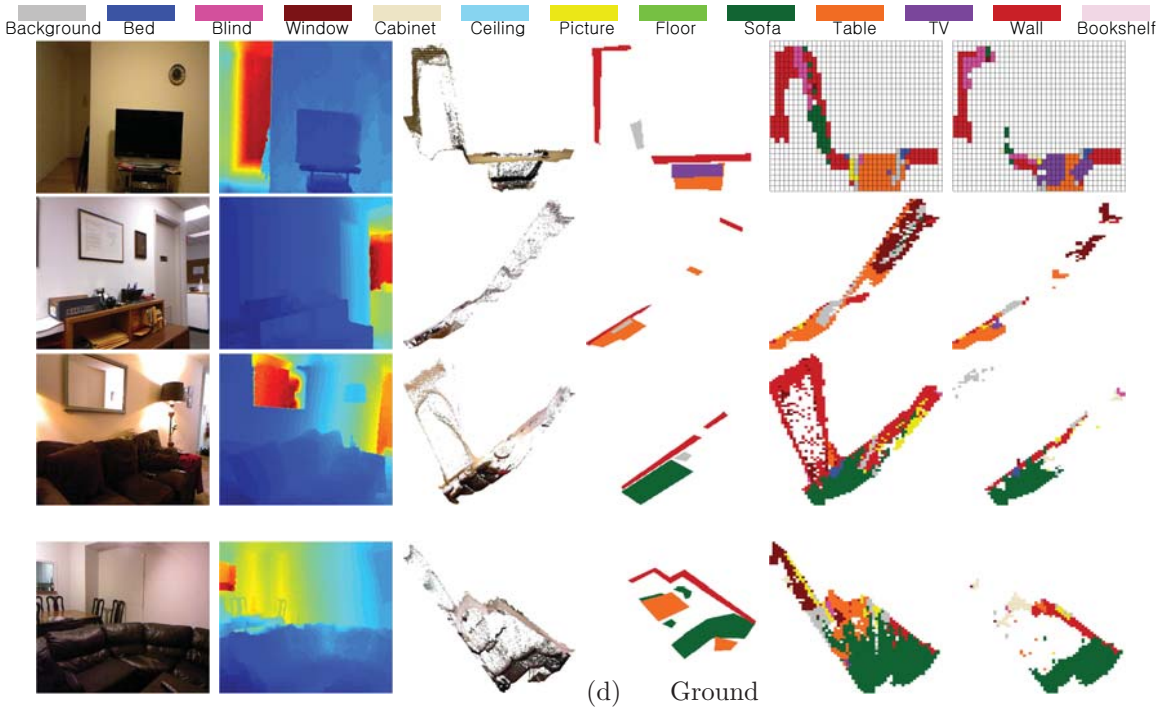


Figure 3.5: Four typical examples show that the 3D geometry of the scene is successfully estimated by solving V-CRF model. Given (a) a RGB image and (b) a depth map, (c) reconstructed 3D geometry (top view) suffers from noise and may not produce realistic scene understanding results. (d) Annotated top-view structured labels (occupied or not, semantic labels). (e) Results from other methods, *e.g.,* [103]. (f) V-CRF achieves labeling and reconstruction results that are closer to the ground truth than [103]. For instance, the empty space (hall) in the first image is successfully constructed with V-CRF, whereas [103] fails. Even with the error due to reflection of the mirror on the third example, V-CRF is capable of reconstructing realistic scenes along with accurate semantic labeling results. We draw a grid to visualize voxels from top view for the first example only.

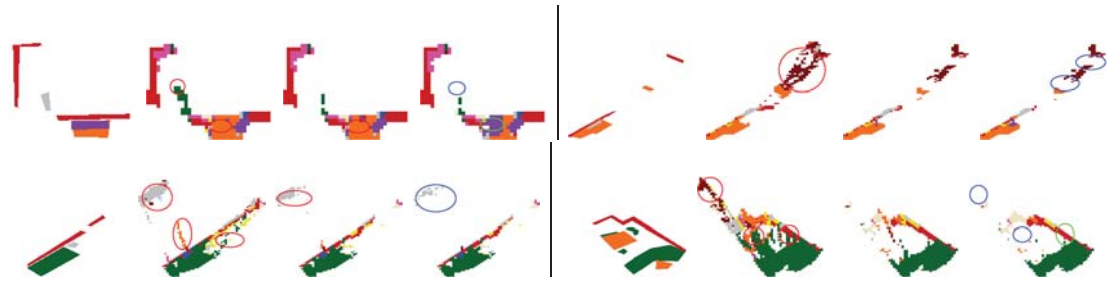


Figure 3.6: Examples show that the iterative inference process improves scene understanding (Sec. 3.5.1). We visualize joint geometric and semantic scene understanding results from its top view. (1,5th column) The annotated top-view ground truth labeling. (2,6th column) V-CRF results after 1st iteration, (3,7th column) after 2nd iteration, (4,8th column) and after 5th iteration. Clearly, as the number of iterations increases, both geometry estimation accuracy and semantic labeling accuracy are improved, as highlighted with blue circles and green circles, respectively. Red circles highlight areas that have been better reconstructed across iterations.

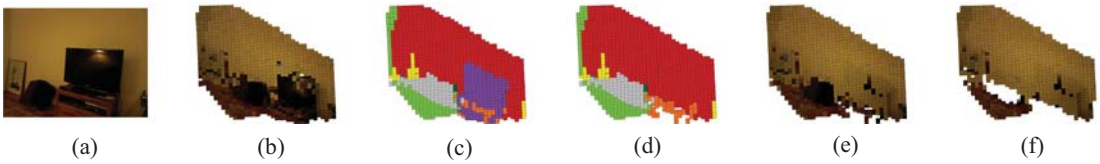


Figure 3.7: (a) RGB image. (b) 3D Reconstruction with V-CRF. (c) Semantic labeling results. (d) Associated voxels for detected 'television' is removed. Note that the region behind the TV is labeled as wall by modeling energy terms for pairwise voxels and planes. (e) As an augmented reality application, TV is removed and voxels are colored with the same color as the adjacent voxels with label 'wall'. (f) All the foreground objects are removed. The occluded region behind the bag is not well reconstructed since there was no plane found behind it. More examples can be found at [101].

CHAPTER IV

Accurate Localization of 3D Objects from RGB-D Data using Segmentation Hypotheses

In this work we focus on the problem of finding objects in 3D from RGB-D images. We propose a novel framework that explores the compatibility between segmentation hypotheses of the object in the image and the corresponding 3D map. Our framework allows to discover the optimal location of the object in 3D using a generalization of the structural latent SVM formulation in 3D as well as the definition of a new loss function defined over 3D space in training. We evaluate our method using two existing RGB-D datasets. Extensive quantitative and qualitative experimental results show that our proposed approach outperforms state-of-the-art as well as a number of baseline methods for both 3D and 2D object recognition tasks.

4.1 Statement

The problem of detecting objects from images that are registered with depth maps (in short, RGB-D images) is receiving increasing interest in computer vision. This is coupled with recent widespread diffusion of depth sensors [94] which allows to accurately measure the distance between the camera and a point in 3D for each image pixel. Researchers have shown that the associated depth information can enhance

detection performances [67, 83] and that, in general, the ability to reason in the 3D physical space provides critical contextual information that does facilitate object detection [2, 54, 65]. However, most of these approaches aim at localizing objects in the image and ignore the problem of estimating object location in the 3D space (we refer to this problem as to *3D object localization*) (Fig. 5.1). This capability is critical in applications related to robotics, object manipulation, safe driving application and video games.

In this work we focus on the 3D object localization problem and propose a new method that is capable of jointly detecting objects in 2D images and the 3D physical space using RGB-D images. Instead of searching for objects in 3D as in [40], which is known to be computationally demanding and prone to false alarms, our approach leverages existing detection methods [36, 99] which identify object proposals in the image by means of bounding boxes. Starting from these bounding box proposals, we introduce a novel framework that explores the compatibility between hypotheses of the object in the bounding box and the corresponding 3D map associated to the pixels within the bounding box. These object hypotheses are generated using hypotheses of object foreground vs background segmentation masks (HFMs) within the bounding box along with the corresponding 3D maps. The intuition is that the ability of combining appearance and corresponding depth values within the HFMs allows constructing more discriminative features for 2D and 3D localization than if such features are extracted from bounding boxes only (Fig. 4.2). Object models are learnt using a latent max-margin formulation whereby the latent variables are the object part locations in 3D. Features are extracted from appearance cues within the HFM and 3D descriptors computed on the associated 3D point cloud. The deformation costs, or penalty costs, for the relative distance between object parts and the object root position, are calculated in 3D space, where a novel efficient 3D matching strategy is proposed. The proposed framework is illustrated in Fig. 4.3. This method

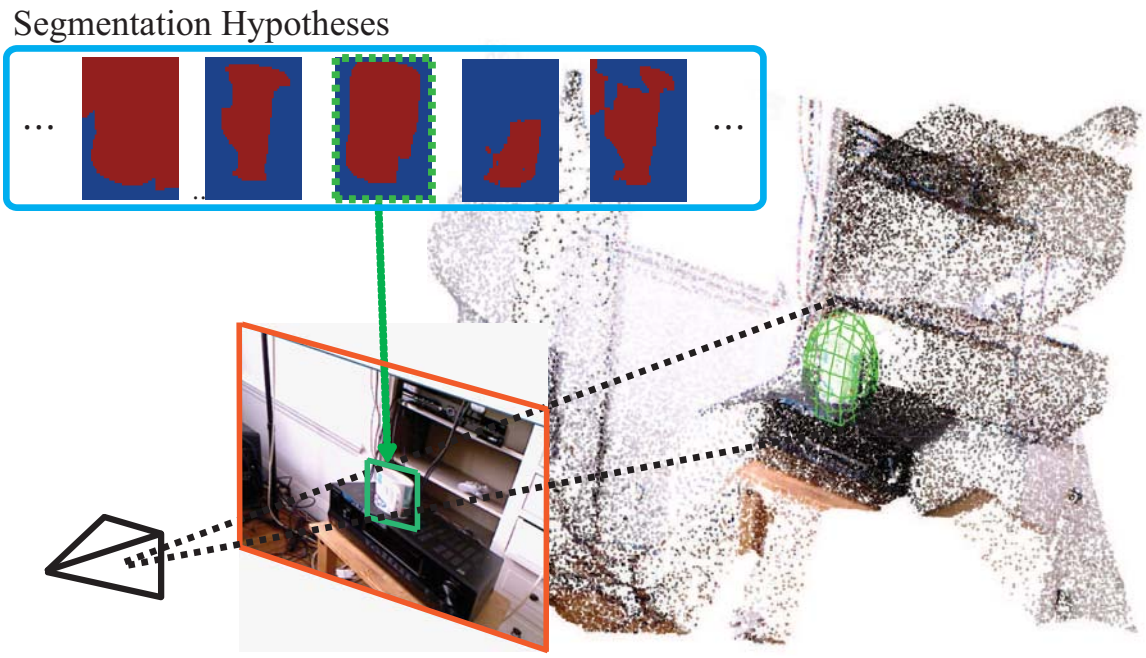
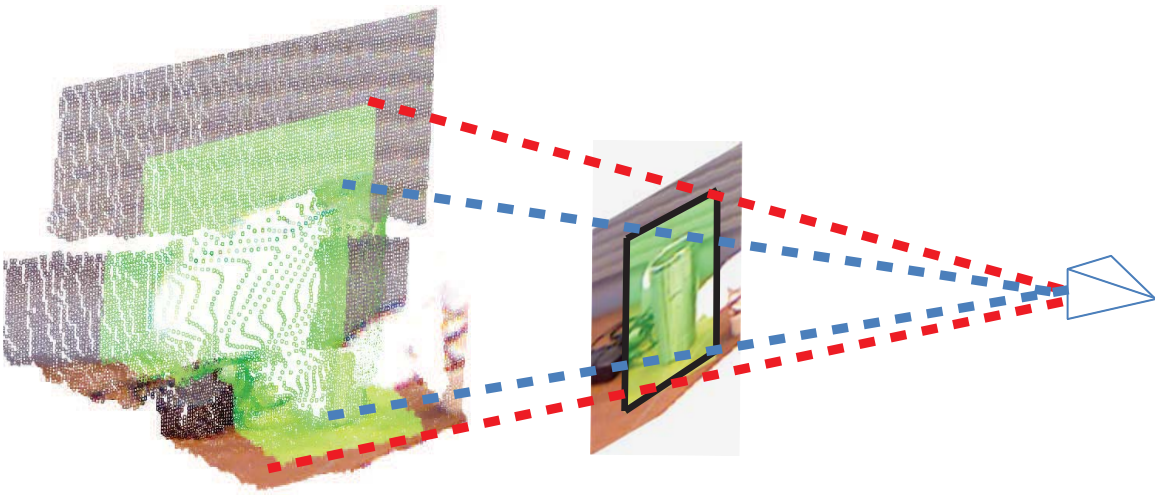


Figure 4.1: In this work we propose a new framework to obtain accurate localizations of objects in 3D by exploring segmentation hypotheses of the object in 2D.

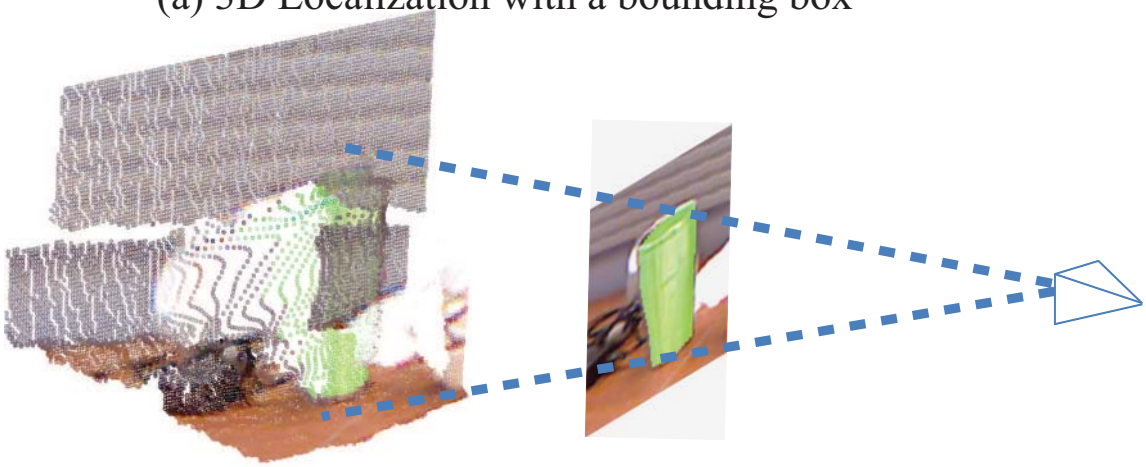
is computationally inexpensive compared to object detection schemes based on sliding bounding cubes in 3D space.

Related work and Contributions. Our overall approach of incorporating depth map to improve image recognition is related to several previous works [40, 67, 74, 83, 97]. For example, [74, 97] built a CRF model using depth map, and showed that RGB-D is useful for indoor scene understanding. [83] used 3D features and obtained improvement in detection performance, and [40, 67] used 3D feature to achieve accurate 2D detection performance. [10] proposed depth map based kernel features for image classification. However, using RGB-D for contextual segmentation or object recognition is still considered as a challenging problem.

The idea of relating detection and segmentation problems in 2D image is connected to works such as [70, 81, 91], where these problems are solved in a joint fashion, which are computationally expensive to process data and to infer the model. However, in this work, we use foreground segments as initial hypotheses efficiently as in [20] and



(a) 3D Localization with a bounding box



(b) 3D Localization with the optimal HFM

Figure 4.2: (a) Detecting an object in 3D (using RGB-D data) from a 2D bounding box is not a trivial problem: the bounding box may include areas of the image that are not related to the foreground object and that correspond to different portions of 3D points in the RGB-D map that are located at completely different distances from the camera. This makes it hard to accurately localize the object position and pose in 3D. (b) In this work we argue that by using segmentation hypotheses for the foreground object (the HFMs), we have the opportunity to identify points in 3D that are only relevant to the object and therefore enable much more accurate 3D localization capabilities.

find out the optimal hypothesis using our novel formulation.

[40] tried to localize objects directly in 3D space using a simple bag-of-words model with linear weights within a branch-and-bound framework. However, the method is

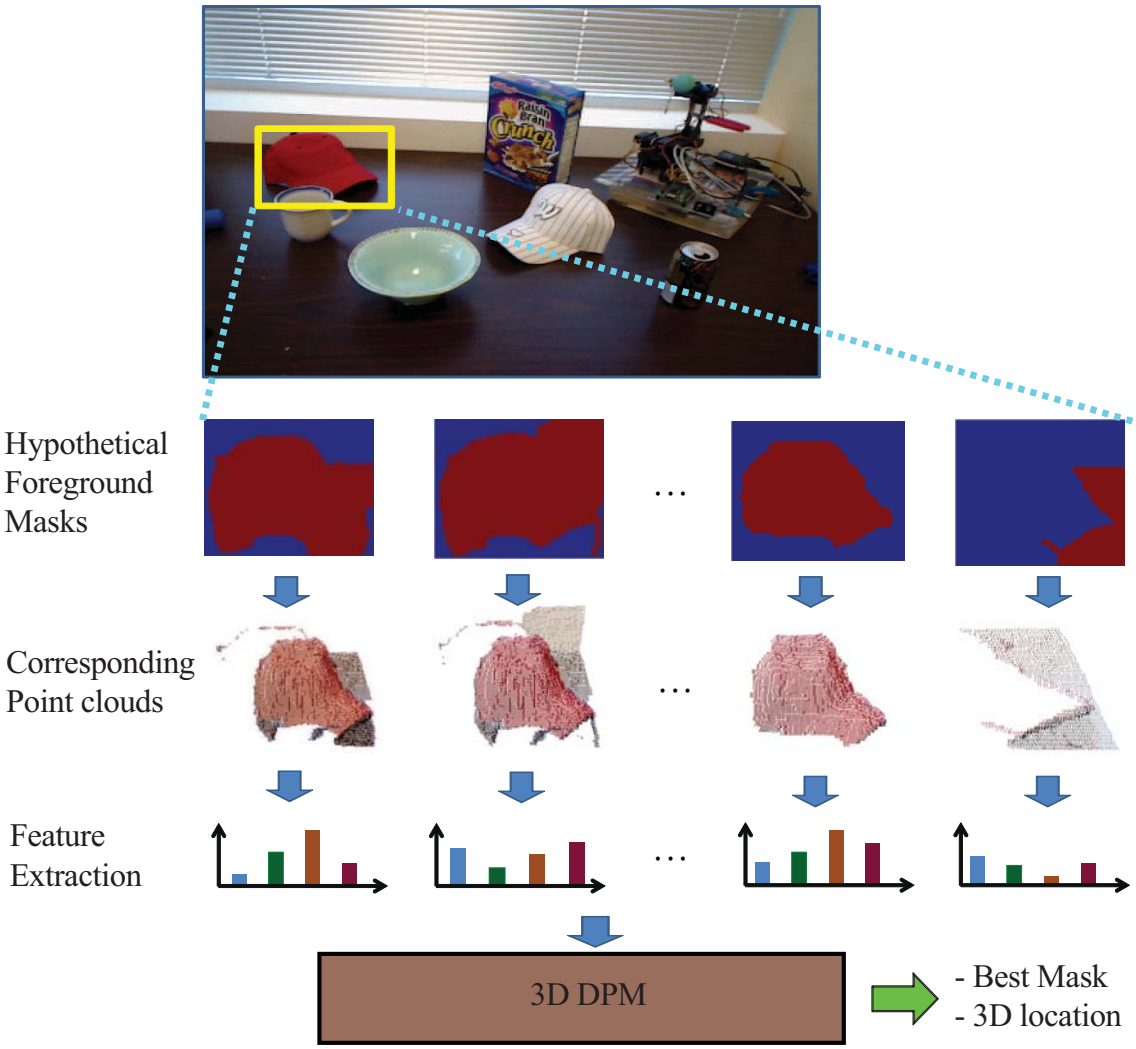


Figure 4.3: This figure shows the process of generating HFM and features from corresponding 3D point clouds. From each bounding box, multiple hypothetical object foreground masks are generated. For each mask, corresponding point clouds as well as features encoding 3D properties of point clouds are generated. From these features, the object's best foreground mask as well as its 3D location are estimated using our structural SVM formulation.

computational expensive since the search space is still large despite the efficiency gain achieved using branch-and-bound.

Our attempt to use a latent structural SVM formulation in 3D is clearly related to [36] as well as to recent work [99] which propose to model an object as collection of 3D parts. [36, 99], however, just seeks to detect objects in 2D images.

Contribution. Our main contributions are four-fold: *i)* we introduce HFM to help extract more descriptive 3D features, leading to a more robust 3D localization (Sec. 4.2.1); *ii)* we propose a novel matching process in 3D, integrating responses from deformable parts in 3D (Sec. 4.2.3). *iii)* we use our structural SVM scheme for joint 3D object localization and selection of the best segmentation hypothesis; finally, *iv)* we provide annotations for 3D object locations on top of existing RGB-D datasets (Sec. 4.3.1).

4.2 Accurate 3D Object Localization with Hypothetical Foreground Masks

In this section, we introduce our framework for accurate object detection and localization in 3D with RGB-D data from a single view. Our main idea is to use HFMs for achieving both efficiency and accuracy in 3D.

4.2.1 Hypothetical Foreground Masks

Bounding boxes. Bounding boxes have been widely used to generate hypotheses of object location in 2D from which features such as HOG can be extracted [36, 123]. The fact that a bounding box contains not only the foreground object but also the portions of the background scene is not necessarily an issue when it comes to object detection. The reason being that the appearance of the background is often correlated to the foreground object (think about a cow sitting on grass) and therefore the combination of the two can enhance object detection. This is much less of a case when RGB-D images are considered and features are extracted from both 2D and 3D point clouds. In such a case, the 3D content associated to portions of a bounding box outside the foreground object can be fairly uncorrelated with the object itself (See Fig. 4.2(a)).

Hypothetical Foreground Mask. In this work we propose to associate each bounding box hypothesis (a HBB) to a set of hypotheses for the foreground object segment (or mask) - the HFM. Specifically, each 2D HBB $y^{b,2D}$ with a height H and a width W is associated with an HFM $y^m \in \{0,1\}^{H \cdot W}$, which is a set of binary variables for all pixels where 1 indicates foreground pixels and 0 is for background. If the mask y^m tightly covers an objects itself, we can map the mask into 3D space as shown in Fig. 4.2(b).

Jointly estimating an accurate $y^{b,2D}$ and y^m is computationally more challenging than estimating $y^{b,2D}$ only. To resolve the problem, we narrow down the searching space for y^m using a discrete top-K object foreground masks extracted using [20]. The typical results of top-K masks are illustrated in Fig. 4.4. To this end, we introduce an auxiliary indexing variable i_m where $y^m_{i_m}$ indicates i_mth mask among K masks.

Feature Extraction From an HFM and the associated HBB, we extract two types of features. First, we extract 3D features from the projected 3D point clouds within the HFM. Designing a 3D feature is out of scope for this work; for our work, we used the modified version of features introduced in [10], which capture 3D properties, i.e., size, norm, etc. Details of our implementation can be found in Sec. 4.3.2. Refer to [25, 71, 113] for examples of possible features that can be used along with our framework. On top of that, HOG features are extracted from a HBB and concatenated.

3D Localization We localize object in 3D space by projecting pixels within the HFM into 3D points to produce accurate localization results. Fig. 4.2 (a) and (b) show localization results from an estimated HBB and HFM, respectively. As the figure shows, when the correct HFM is used, the corresponding 3D point cloud enables much more accurate localization results than if an HBB is used. In Sec. 4.3.7, we quantitatively and qualitatively show that the proposed scheme significantly improve the 3D localization performance.

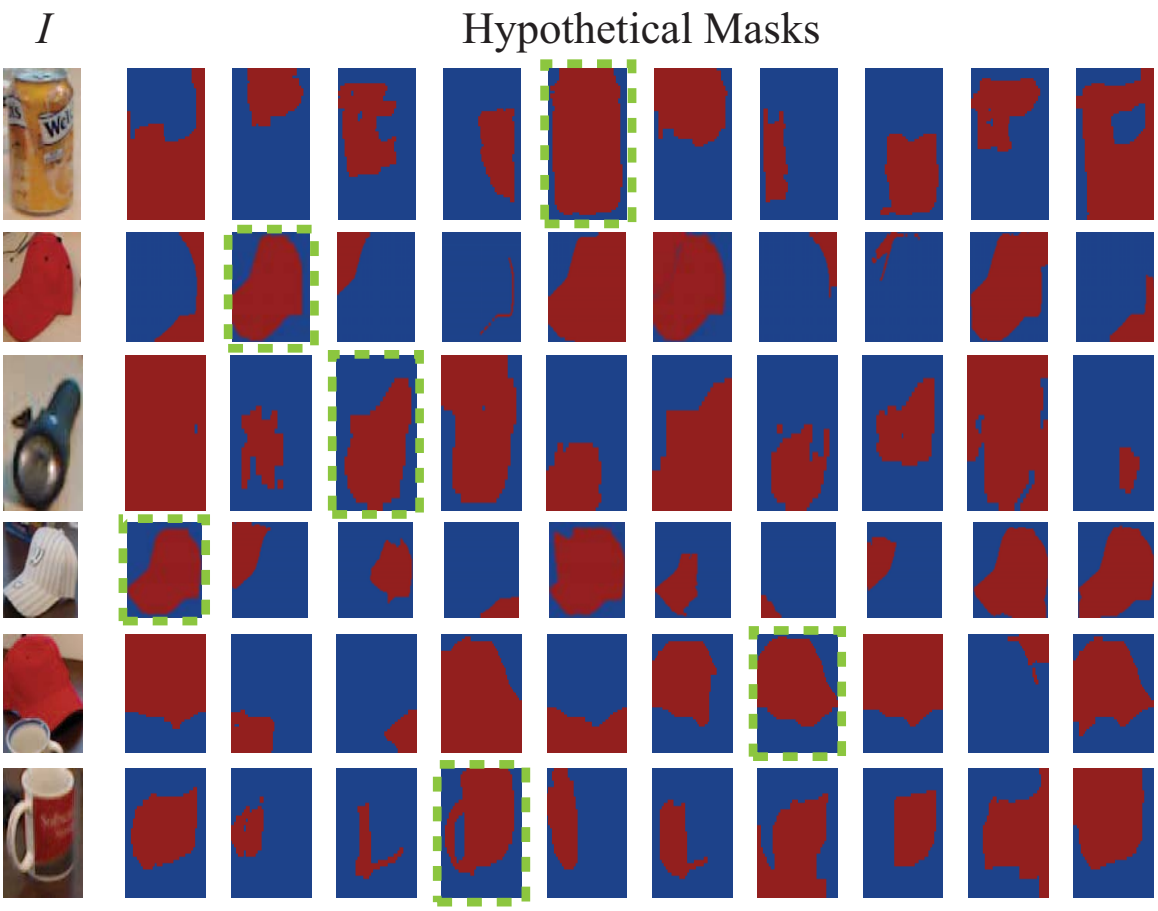


Figure 4.4: The first column is the RGB image inside bounding box. Remaining columns show top K foreground segmentation hypothesis when $K = 10$. The hypotheses highlighted with green lines indicate the segmentation which is closest to the ground truth.

4.2.2 Part Based Model in 3D

Inspired by the deformable part based model (DPM) presented in [36] which estimates object bounding boxes and their latent part locations in the 2D image, our framework determines the optimal $3D$ location of the object $y^* = (y^{b,2D*}, y^{m*}_{i_m})$ as well as its parts location h^* in 3D as $(y^*, h^*) = argmax_{(y,h)} \langle \beta, \Psi(I, y^{b,2D}, y^m_{i_m}, h) \rangle$. The feature vector $\Psi(I, y^{b,2D}, y^m_{i_m}, h)$ concatenates features for M components of the mixture model, which encode 2D and 3D appearance cues, 3D distances between root and part filters and a offset value. The linear classifier β is learned using the

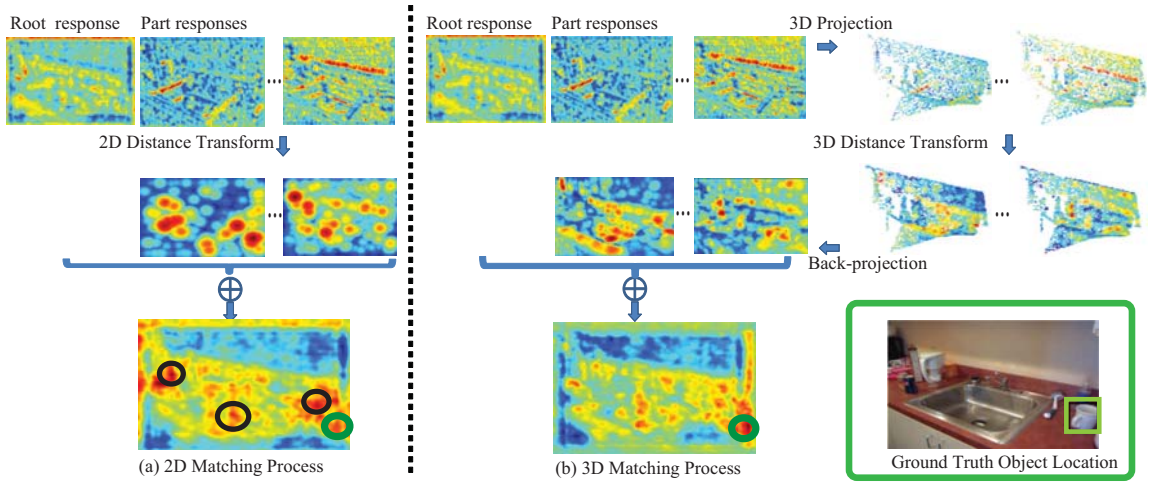


Figure 4.5: This figure shows the process of 3D matching (Fig.(b)) compared with matching in 2D (Fig.(a)). By applying 3D matching, possible false alarms (black circles in Fig.(a)) can be suppressed if 3D distances between root and part filters are large. For 3D matching, part responses are firstly mapped into 3D space, and 3D distance transform is applied to efficiently calculate deformation costs between root and part filters. Details for the 3D matching can be found in the text.

Structural LSVM framework (Sec. 4.2.4).

4.2.3 3D Matching

The procedure to estimate the root and part location is referred to as *matching* [36], which takes into account the 2D Euclidean distance between filter locations [34], as shown in Fig. 4.5(a). On the other hand, our framework discovers the best 3D root and part locations, and this process is referred to as *3D matching*. By looking at 3D distance between root and part filters, this process suppresses false alarms in identifying the object parts if 3D distance between root and part is large, even if they appear close in 2D image. As a result, possible false alarms in results of 2D matching (Fig. 4.5(a)) are removed by using 3D matching (Fig. 4.5(b)).

For 3D matching, first, we project response map into the 3D point cloud by associating a confidence value of a pixel to its corresponding point in 3D. Then, we define a score function which is obtained as the summation of the root and parts

responses, with respect to their deformation costs in 3D. This score function gives a highest score at its optimal location and is expressed as follows:

$$\begin{aligned} \text{score}(x_0, y_0, z_0, l_0) &= R_{0,l_0}(x_0, y_0, z_0) \\ &+ \sum D_{i,l_0-\lambda}(2(x_0, y_0, z_0) + v_i) \end{aligned} \tag{4.1}$$

$R_{i,l}(\cdot)$ is filter responses projected into 3D space for a part i at scale l. The variable i indicates part index if $i > 0$, or it indicates root if $i = 0$. v_i is the relative anchor position the part i. λ is the scale difference between root and part filters. The transformation $D_{i,l}(\cdot)$ is used to allow for spatial uncertainty in parts location in 3D,

$$D_{i,l}(x, y, z) = \max_{dx,dy,dz} (R_{i,l}(x + dx, y + dy, z + dz) - d(x, y, z)) \tag{4.2}$$

where $d(x, y, z) = d_i \cdot \phi(dx, dy, dz)$ is the weighted Euclidean distance.

Calculating $D_{i,l}(\cdot)$ over 3D space is computationally expensive, which takes $O(N^3)$, where N is the size of the searching space for 1D. Note that [34] showed that this transformation can be efficiently calculated in case of 1D for a quadratic cost function. For 3D matching, our cost function is 3D Euclidean distance, which is a quadratic function over (x, y, z). Thus, we can efficiently obtain the transformation in 3D by iteratively applying the 1D distance transform as follows:

$$\begin{aligned} D_{i,l}(x, y, z) &= \max_{dx',dy'} (R_{i,l|dz'}(x + dx, y + dy) - d_{|dz'}(x, y)) \\ &= \max_{dx'} (R_{i,l|dy',dz'}(x + dx) - d_{|dy',dz'}(x)) \end{aligned} \tag{4.3}$$

which makes computational time into $O(N)$.

Once the root location is found in 3D, parts locations also can be found by looking up the optimal displacements, similar to the 2D case [36].

4.2.4 Structural LSVM in 3D

To train model weights β, we propose to use Structural Latent SVM (StLSVM) framework [69] by considering 3D objects locations as their labeling space. This can improve the precision of decision boundaries of trained classifier since it penalizes inaccurate 3D localization predictions during the training process. In the following, we describe how the labeling space in 3D is formulated, and also introduce an appropriate loss function.

Labeling Space with Foreground Mask and Associated 3D Ellipsoid. Our training data is equipped with object class label y^l and the object foreground mask y^m, *i.e.,* $y = (y^l, y^m)$. To help associate the mask with 3D locations, we use y^s which is equivalent to y^m with different parametrization; $y^s = \{(u_1, v_1), ..., (u_S, v_S)\}$ is indicating pixels of the object foreground mask where $y^m(u, v) = 1$. S is the number of pixels belonging to the foreground region. $y^l \in \{-1, 1, ..., C\}$ is the class of the depicted object or -1 for background. The location of 2D bounding box, $y^{b,2D}$, is determined from y^s, by retaining the minimum and maximum indices over the image axes u and v. On top of that, we obtain 3D object location by projecting y^s to point clouds $y^{s,3D}$ as follows.

$$\begin{aligned} y^{s,3D} &= g(y^s, \text{Depth}, \text{Camera}) \\ &= \{(u'_1, v'_1, z'_1), ..., (u'_S, v'_S, z'_S)\} \end{aligned} \tag{4.4}$$

where $g(\cdot)$ is the projection function given the depth map and camera parameters. (u'_i, v'_i, z'_i) is the 3D location of a point cloud. We use 3D ellipsoids capturing point cloud $y^{s,3D}$ to identify objects in 3D space. 3D ellipsoids are defined using a $9-$ parameter model. As we will discuss in Sec. 4.3.1, ellipsoids are more convenient

(than bounding cubes) for annotating objects in 3D. In specific,

$$\begin{aligned} y^{b,3D} &= \text{Ellipsoid}(y^{s,3D}) \\ &= [c_x, c_y, c_z, v_1, v_2, v_3, d_1, d_2, d_3] \end{aligned} \tag{4.5}$$

where $\{c_x, c_y, c_z\}$ is the center, $\{v_1, v_2, v_3\}$ major axes, and $\{d_1, d_2, d_3\}$ are radii of the ellipsoid.

Training. The training data is $\{(I_i, y_i)\}_{1,\dots,N}$, where $\{I\}$ is the set of images, and $\{y_i = (y_i^l, y_i^s)\}$ are labels. The model learns the parameter by solving the following latent max-margin optimization problem,

$$\begin{aligned} \underset{\beta,\xi}{min} \quad & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i \\ s.t. \quad & \forall i, I_i, \bar{y} \neq y_i : \underset{h_i}{max}\,\langle \beta, \Psi(I_i, y_i, h_i)\rangle \\ & - \underset{h}{max}\,\langle \beta, \Psi(I_i, \bar{y}, h)\rangle \geq \Delta(y_i, \bar{y}) - \xi_i \end{aligned} \tag{4.6}$$

where $\bar{y}$ is the most violating prediction. Note that, since y contains information of 3D ellipsoid location, it is able to take the localization accuracy into account for designing $\Delta(y_i, \bar{y})$.

Finding the Most Violating Sample. Obtaining the most violating sample $\bar{y}$ is not easy because $|\bar{y}^m| = 2^{H \cdot W}$, where H and W are the bounding box height and width, respectively. With a set of HFMs $\{y_{i_m}^m\}$, we resolve this issue by efficiently finding i_m which gives most violating $\bar{y}_{i_m}^m$ among $\forall i_m \in 1 \sim K$.

Loss Function in 3D. We design the loss function $\Delta(y_i, \bar{y})$ depending on both 2D and 3D localization accuracies. Similar to [99],

$$\Delta(y_i, \bar{y}) = \begin{cases} 0, & \text{if } y_i^l = \bar{y}^l = -1 \\ 1 - [y_i^l = \bar{y}^l]\frac{A(y_i \cap \bar{y})}{A(y_i \cup \bar{y})}, & \text{otherwise} \end{cases} \tag{4.7}$$

$A(y_1 \cap y_2)$ and $A(y_1 \cup y_2)$ are the intersection and union of two object locations, respectively. We calculate the two values as a weighted sum of intersection and union in 2D and 3D as follows:

$$A(y_1 \cap y_2) = w_1 A(y_1^{b,2D} \cap y_2^{b,2D}) + w_2 A(y_1^{b,3D} \cap y_2^{b,3D}) \tag{4.8}$$

$$A(y_1 \cup y_2) = w_3 A(y_3^{b,2D} \cup y_2^{b,2D}) + w_4 A(y_1^{b,3D} \cup y_2^{b,3D}) \tag{4.9}$$

where $A(y_1^{b,2D} \cap y_2^{b,2D})$ is the intersecting area between two 2D bounding boxes, and $A(y_1^{b,3D} \cap y_2^{b,3D})$ is the intersecting volume between two 3D ellipsoids[1]. The union $A(y_1 \cup y_2)$ is calculated in a similar fashion. During the experiments, we set $w_{1,2,3,4} = 0.5$.

4.3 Experiments

In the following, we describe the experimental settings and results. We evaluate our framework using two datasets (Washington RGBD (WRGBD) and Berkeley 3D Object (B3DO) datasets) and provide annotations of 3D object locations for learning and for providing ground truth information.

4.3.1 Annotation

While many 3D datasets [42, 67, 83, 97] have appeared in last few years, none of them provide annotations that allow to localize objects in 3D (with the expection of [42]). For example, [67, 83] annotated locations of objects in 2D space with 2D bounding boxes, without 3D location. Objects in [97] contain pixel-wise labeling for object instances, but objects are often too small or contain severe occlusions or truncations. [42] include range data along with accurate location with 3D cubes, the range data for training set is not provided.

[1]See the supplementary material for the method to calculate intersecting volume between two ellipsoids.

In our work, we parameterize object location using 3D ellipsoids. 3D ellipsoids are good to capture the size with 3 major axes of the object, and also describe objects' location in 3D space accurately. Also, they are easy to use for annotation. At that end, we have created an easy and efficient labeling tool. Using this tool, the annotator can simply draw a polygon capturing object foreground, the 3D points corresponding to the pixels enclosed by the polygon are used to calculate the centroid and the principal axes of the tightest ellipsoid enclosing such 3D points. Principal axes are calculated using PCA on the point cloud. Statistics related to our annotated ellipsoids and its comparison with other statistics can be found from a supplementary material. Typical examples of the annotation results can be found in the second column of the Fig. 4.11. In our framework, overlap ratio between ground truth and estimated ellipsoids are used to calculate the loss function for training StLSVM model, as well as for evaluating 3D localization performance.

4.3.2 Implementation Details

For the experiments with the B3DO dataset, we concatenated HOG features calculated from deformable parts [36] with 3D features proposed in [10]. For the experiments with the WRGBD dataset, we further concatenated HOG features extracted from depth map as proposed by [83].

4.3.3 Foreground Mask Accuracy

There is a trade-off between the computational complexity and the number of hypothetical masks. By using a larger number of hypotheses, there is a higher chance to pick up the correct one. This is at the expense of the added computational time that is required to calculate features and apply the object model.

In Fig. 4.6, we measured the accuracy as function of the number of HFMs. Accuracy is measured using the best F-measure criterion $F = \frac{2RP}{P+R}$, where P and R are

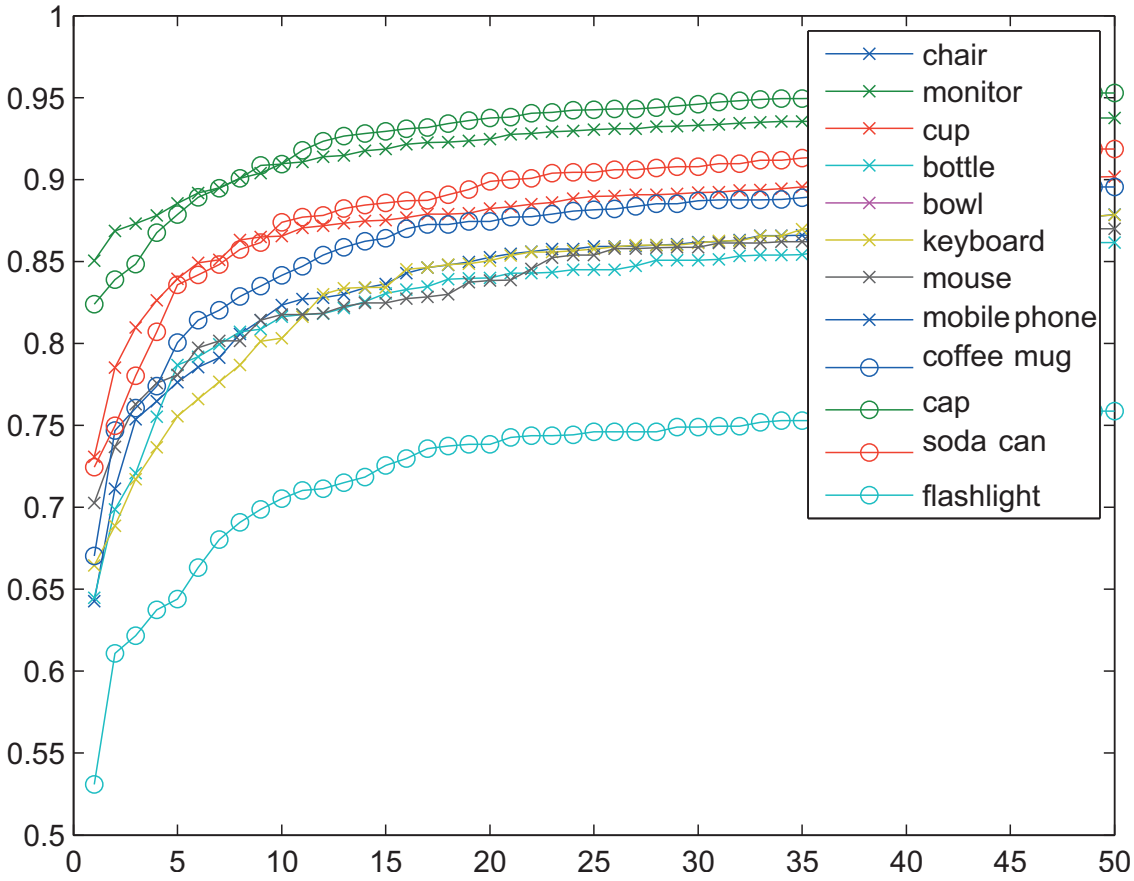


Figure 4.6: This figure shows F-measure as a function of the number of HFMs averaged over each object class for both B3DO (marked with 'x') and WRGBD (marked with 'o') datasets.

the precision and recall of pixels in a segment relative to the ground truth [1]. From the curve, we can see that the accuracy increases as function of the number of HFMs. Notice that when this number is greater than 10, the performance gain becomes negligible. In the experiments we discuss next, we set the number of hypothetical masks K as 10.

4.3.4 Berkeley 3D Object Dataset

We first evaluated our proposal with the Berkeley 3D Object Dataset (B3DO) [67]. Among available object classes in the dataset, we tested 8 classes for which [67]

evaluated the performance of 2D localization. 3D localization was not tested in [67], so we use several baseline methods to evaluate our framework.

4.3.5 3D Detection Performance

Accuracy is computed by using 3D ellipsoids as discussed in Sec. 4.3.1. Similar to Pascal Challenge criteria, localization is counted as correct if the overlapping volume between estimated ellipsoid and the ground truth ellipsoid is more than a threshold. Otherwise, it is counted as wrong. In our experiments, we set the threshold to be 25%.[2] We compare our method with the following baselines:

DPM+FillMask. For a detected 2D bounding box, we project all the pixels inside that bounding box. The ellipsoids are generated to capture all the corresponding to 3D points.

DPM+1stMask. Among K hypothetical masks, we choose the top-ranked mask from [20] as a foreground mask. The score corresponding that mask is used to evaluate the detection.

DPM+SizePrior. From a depth map and a bounding box, object location, width and height of the object are estimated. The depth of the object is set to a mean value that is computed by averaging out ground truth object depths for each class in the training set.

Results. Fig. 4.7 shows the average precisions of 3D localization results of proposed method, compared with all the baselines methods. Our method achieved the best performance for 7 out of 8 categories, and on average, it attains at least 6.2% higher average precision than baseline methods. For the class *cup*, DPM+SizePrior and our method achieve similar performance. The reason may stem from the fact that since there is small variance in the size of objects in the *cup* class, DPM+SizePrior can successfully capture its 3D location well. On the other hand, for classes having

[2]While 2D detection often use 50% as a threshold, 3D localization is more challenging and 25% is a reasonable threshold for evaluation. For more details, see the supplementary material.

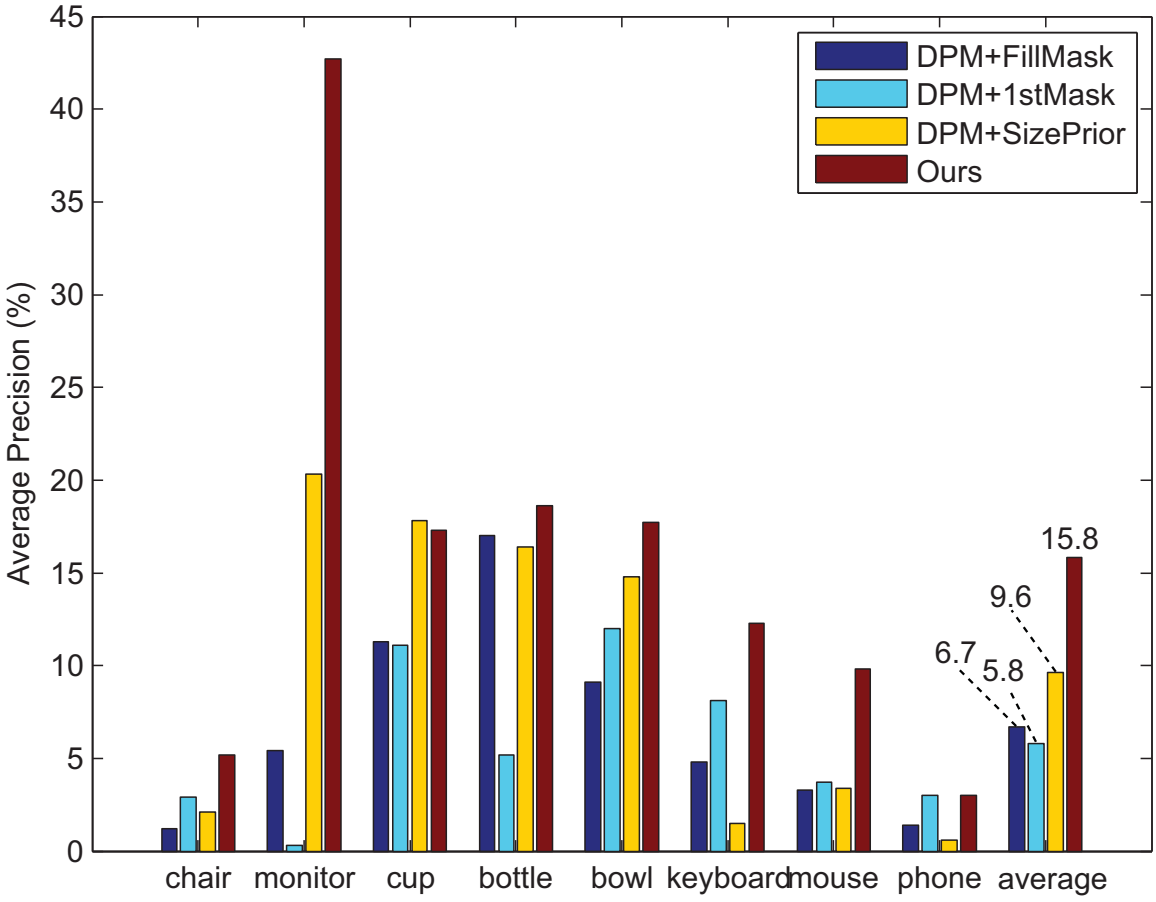


Figure 4.7: Average precisions of 3D object localization for 8 classes in B3DO dataset. Our method achieves best results compared the a number of baselines.

large variances in their depth due to different poses (for example, *monitor* or *keyboard*), our method works better than all baselines. Typical 3D localization results can be found in Fig. 4.11.

4.3.6 2D Detection Performance

We also show that our method improves 2D detection accuracy. Fig. 4.8 shows the average precisions of the detection results in 2D using the B3DO dataset. We compare our performances with DPM [36] and two methods proposed in [67]. The first method is called *pruning*, where detected results are pruned out if the approximated object size (bounding box diagonal times mean depth) is different from the statistics of the

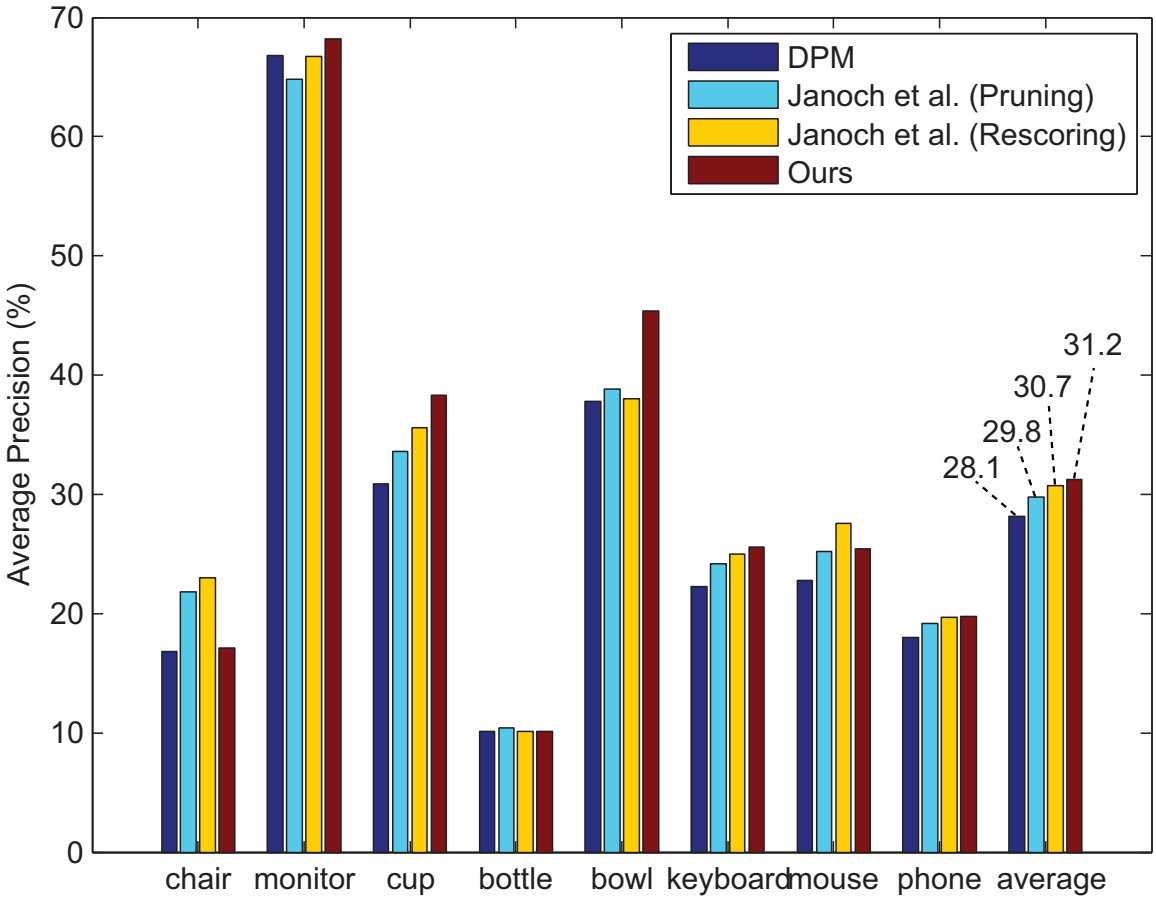


Figure 4.8: Average precisions of 2D object localization from DPM, two methods proposed in [67] and our method on B3DO dataset. Note that our proposed method consistently achieves better average precision over [67].

dataset. The second method is called *rescoring*, in which linear SVM is trained with additional features of approximated object size [67].

Note that we achieve better results for 6 out of 8 categories. This confirms that using HFM and associated 3D features is beneficial even for a 2D detection task. Notice that there is no improvement for the *chair* category. This may be due to severe occlusions that occur for the *chair* category in the dataset and that are not well characterized using our model.

4.3.7 Washington RGB-D Object Dataset

We also evaluated the proposed method using the Washington RGB-D Object Dataset (WRGBD) [83]. Note that in order to make the comparison with [83] fair, the features are extracted from RGB, depth map as well as estimated object size as in [83].

3D Detection Performance Fig. 4.9 shows the average precision for 4 classes, *coffee mug*, *cap*, *soda can*, and *flashlight*, in the WRGBD dataset. Again, we achieve best accuracy compared to the baseline methods discussed in Sec. 4.3.5. We notice that the objects in this dataset have small variance in their size and pose, so that the baseline DPM+SizePrior already achieves a good 3D localization performance. Note that our framework further improves the accuracy by roughly 3%.

2D Detection Performance Fig. 4.10 shows average precisions of our method, and of baseline methods. Although, as discussed earlier, the features used for baseline methods already contains information extracted from both RGB and depth map, our framework achieves the best performance compared to them.

4.4 Conclusions and Future work

In this work we proposed a new approach for localizing objects in 3D using RGB-D images. We explored the idea of using segmentation hypotheses for the foreground object to guide the process of accurately localizing the object in 3D. Extensive experimental analysis has demonstrated our theoretical claims. Directions for future work include the ability to integrate segmentation hypotheses in both 2D and 3D.

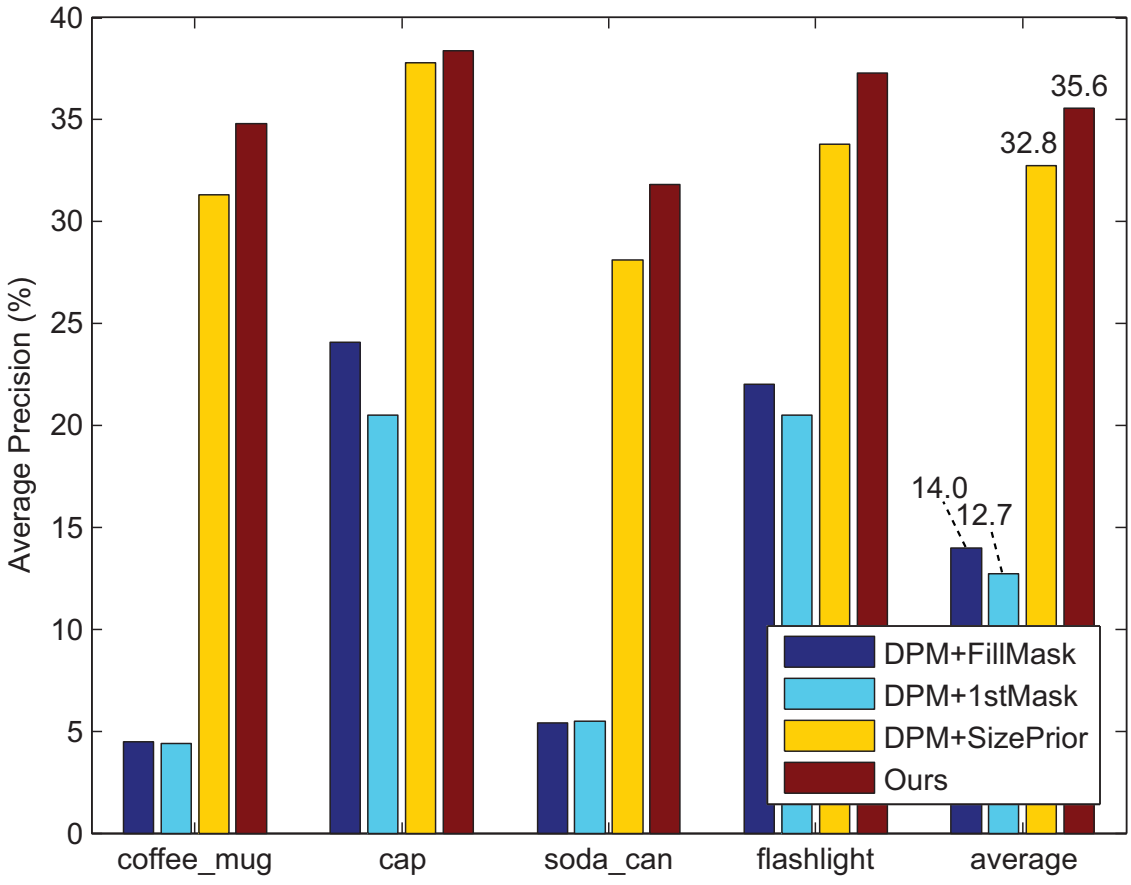


Figure 4.9: Average precisions of 3D object localization in the WRGBD dataset. Our method achieves best results compared with all baselines.

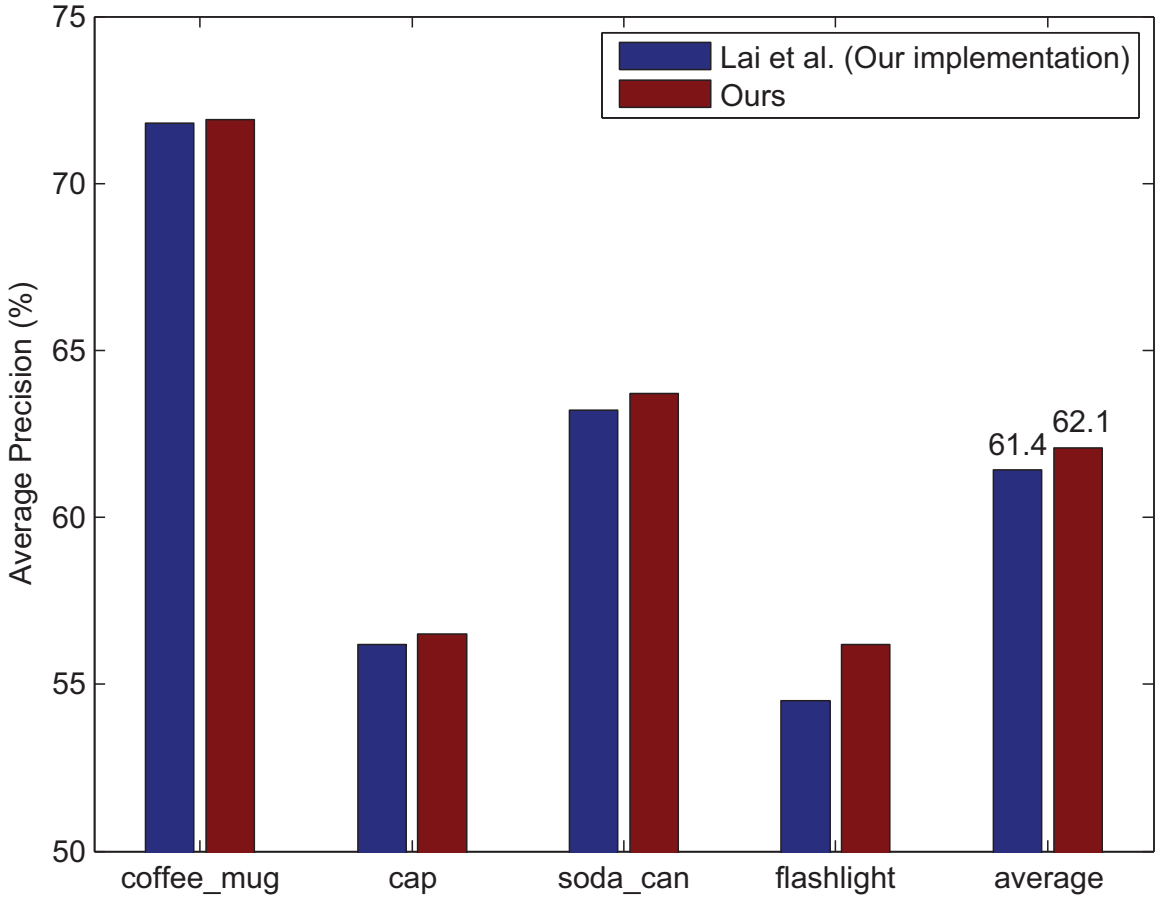


Figure 4.10: Average precisions of 2D object localization from DPM (with features proposed in [83]) and our method on the WRGBD dataset. Our proposed method consistently achieves better average precision over [83].

Figure 4.11: This figure shows typical examples of object localization in 3D obtained using the proposed model and baseline methods. Each column represents ground truth bounding boxes in 2D, ground truth bounding boxes in 3D, 3D localization results using 3 baseline methods, and 3D localization results using our method, respectively. The localization results are drawn with black ellipsoids and green is used for ground truths. First four rows shows good examples. Notice the ellipsoids estimated by our framework is very close to ground truth ellipsoids, whereas baseline methods give less well localized ellipsoids. The last row shows failure cases. More typical examples can be found in the supplementary material.

CHAPTER V

Hierarchical Classification with Sparse Approximation

Using image hierarchies for visual categorization has been shown to have a number of important benefits. Doing so enables a significant gain in efficiency (e.g., logarithmic with the number of categories [48, 93]) or the construction of a more meaningful distance metric for image classification [98]. A critical question, however, still remains controversial: would structuring data in a hierarchical sense also help classification accuracy? In this work we address this question and show that the hierarchical structure of a database can be indeed successfully used to enhance classification accuracy using a sparse approximation framework. We propose a new formulation for sparse approximation where the goal is to discover the sparsest path within the hierarchical data structure that best represents the query object. Extensive quantitative and qualitative experimental evaluation on a number of branches of the Imagenet database [27] as well as on the Caltech-256 [48] demonstrate our theoretical claims and show that our approach produces better hierarchical categorization results than competing techniques.

5.1 Statement

Recent advances in computer vision and machine learning have enabled the design of recognition methods that are capable of classifying images into large number of visual categories (typically, hundreds) [23, 30, 49, 84]. In one of the current paradigms for image categorization, image classes are organized in a flat structure and the problem is to discover the class (among all those in the flat structure) that best represents (in term of a distance function) the visual content of a given query image.

Recently, researchers have explored the idea of organizing visual data in a hierarchical structure rather than in a flat one. This paradigm addresses some of the limitations of the flat structure: i) it allows for a significant gain in efficiency, typically logarithmic with the number of categories, as addressed by Marszalek and Schmid [93] and Griffin and Perona [48]; ii) it enables the construction of a more meaningful distance metric for image classification; iii) it echoes the way how humans organize data, as addressed by Palmer [98]. However, a critical question still remains controversial: would structuring data in hierarchical sense also help classification accuracy? Up to date there is no definite answer to that question. For instance, top-down classification schemes (applied on hierarchical structures) proposed by Marszalek and Schmid [93] and Griffin and Perona [48] have produced inconclusive evidence as for whether hierarchy has a beneficial effect on classification accuracy. Classification methods based on Hierarchical Support Vector Machines can be used to trade off accuracy against speed as in Griffin and Perona [48] or employed to increase classification accuracy as originally proposed by Tsochantaridis *et al.* [121] and utilized for image classification, as suggested by Binder *et al.* [7]. Although [7] has shown promising results, it is computationally very demanding as the number of categories becomes larger than 30$\sim$50. Finally, methods based on combining models from different levels of the hierarchy proposed by Zweig and Weinshall [130] have also shown positive results but are yet to be validated on deeper and larger hierarchical structures.

In this work we attempt to address the issues discussed above and show that the hierarchical structure of a database can be successfully used to enhance classification accuracy using a sparse approximation framework. The key idea is to introduce a distance function that takes into account the hierarchical structure of the visual categories (Figure 5.1) and to identify two images to be similar if they share a similar path in the hierarchy. We show that this distance function (or similarity metric) is equivalent to the Hamming Distance (HD) for vectors that encode the hierarchy. This allows us to cast the categorization problem as the one of discovering the category in the tree structure that has the smallest HD from the query category label. We solve this problem via sparse approximation and introduce a new formulation of the sparse approximation problem which we call *hierarchical* sparse approximation. In the typical sparse approximation problems, [21, 120, 126], a query image can be identified as the sparsest representation over the set of training images, as proposed by Wright *et al.* [126] or basis functions, as proposed by Mairal *et al.* [90] for all object classes; that is, the sparsest solution is one (or a combination of a few) image out of *all possible* images in the dataset. We call this the *flat* sparse approximation problem. The key novelty of our approach relies on the idea that the sparse representation is not constructed over a *flat* structure of object classes (as in the classic sparse sensing problem) but rather by enforcing that the solution must be one (or a combination of a few) path out of all possible paths on a given hierarchy of object classes (training set). Moreover, classification accuracy is measured in hierarchical sense (that is, by considering the HD between the query path and the ground truth one). Since our method relies on the sparsity of the representation, our approach is suitable for large scale classification problems; i.e., the conditions underlying the sparsity assumptions are best verified when the dataset is large and distribution of visual categories is diversified. In this work we present sufficient conditions under which our hierarchical sparse formulation can be used with success and small error bounds are guaranteed.

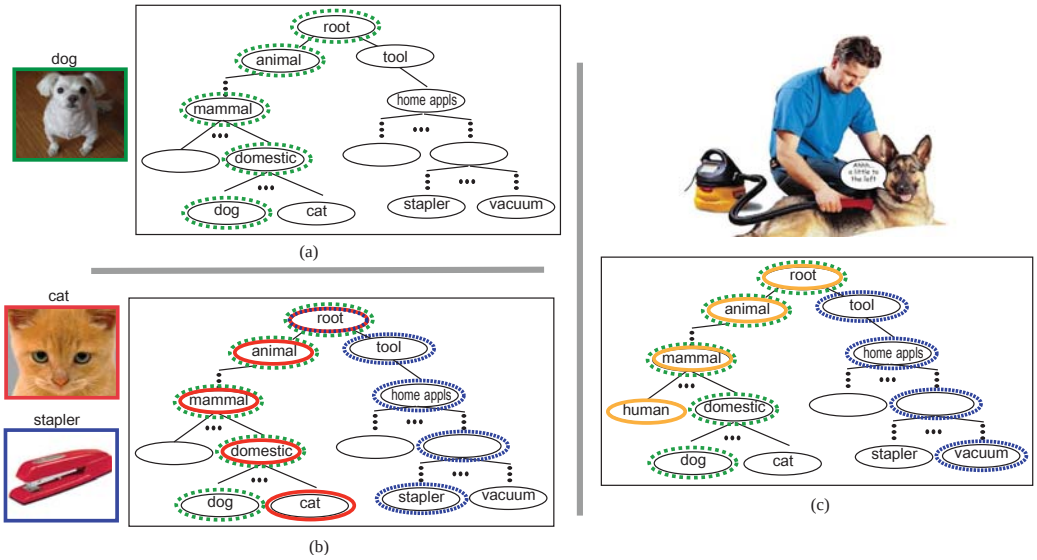


Figure 5.1: (a) Organizing images in a hierarchical structure (tree) enables more descriptive methods for characterizing images: the image of a dog can be described by class labels associated to each node of the (green) path in the tree. (b) Misclassifying a dog with a cat is not as bad as misclassifying a dog with a stapler. If data are organized in a tree, it is possible to relate object classification errors with objects with locations in the tree. For instance dog, cat and stapler categories are associated with the green, red and blue paths (respectively) in the tree. The error in misclassifying a dog with a cat can be measured as the Hamming Distance (HD) between the corresponding paths. HD captures the similarity between two paths in the tree (see Section 5.3.1 for details). The HD is 1 in this case. Note that misclassifying a dog with a stapler leads to a larger HD (that is, 5). (c) It is desirable to classify multiple objects at the same time. If an image contains a dog, a human and a vacuum, our algorithm can discover three paths (green, orange and blue respectively) in the tree, one for each query object.

Furthermore, a crucial property of our classification framework is that it is capable of classifying multiple object categories at the *same time* if more than one (dominant) object appears in the query image (Figure 5.1 (c)).

We have carried out extensive quantitative and qualitative experimental evaluation on a number of branches of the Imagenet database [27] as well as Caltech-256 [49]. Each branch comprises hundreds of visual categories organized in the hierarchical structure. All the experiments demonstrate that our hierarchical approximation framework yields much better hierarchical classification accuracy over *flat*

sparse approximation. Evaluation was carried out by comparing average precision measured in terms of HD as well as by measuring the actual classification accuracy at each level of the hierarchy. Our method achieves a performance increase ranging from 5% to 10% for the most critical levels of the hierarchy. Additional experiments on multi-category classification also show very promising results.

The rest of this work is organized as follows. In Section 5.2, we will briefly review how sparse approximation can be applied to image classification problem. The formal definition of hierarchical classification and our proposed embedding is provided in Section 5.3.1. A number of experiments are carried out to validate our scheme in Section 5.4. Finally, we summarize our work in Section 5.5.

5.2 Image Classification using Sparse Approximation

In this section, we describe our image representation and introduce the basic formulation of the flat image classification problem based on sparse approximation. We assume a database of images is available. Furthermore, we assume that such a database comprises a large number of categories and each category has a large number of image instances. We assume that each image has a dominant object instance with some level of background clutter as in Caltech-256 [49] or the ImageNet [27]. In classification, we assume that the query image (with unknown category label) contains one (or multiple) dominant object(s) whose category label is represented by the dataset. Of course, the query object instance itself is not necessarily included in the dataset. The classification problem can be solved by seeking, among all the images (object instances) in the database, the one that is closest to the query object(s). The category such image belongs to is the classification result. If the query image contains multiple dominant objects, the classifier must return multiple category labels associated to all of the dominant objects in the query image.

5.2.1 Object representation and distance function

Assessing whether an image is "close" to another one relies on the construction of a distance function which depends on the way how the visual content of an image is represented. Following a common representation used in computer vision, we describe an image using a normalized histogram of codewords (i.e., the bag of words representation, also named BOW) [23] or, equivalently, a histogram capturing a spatial pyramid of codewords [47, 84]. In either cases, we denote such histogram by a vector x. Codewords are drawn from a learnt dictionary of vector quantized features as described in [23, 47, 84]. The size of the dictionary is denoted by K. Thus x is a column vector of size K, if we use a simple histogram of codewords to represent the image. Notice that other types of representations are also possible. The similarity between two images represented by x_i and x_j can be measured by computing the l_n norm distance between x_i and x_j, where n can be 0, 1, etc. Similar images will have small distances.

5.2.2 Model matrix

Let us stack all the histograms of images in the database as columns of the matrix H. Thus, H will be $K \times N$, where N is the number of images in the dataset. We call this matrix H the *flat model matrix*. Under the assumption that the database is sufficiently large, any query image can be represented as a superposition of one or more images in the training data with small error e such that $x = Hm + e$. Note that $N \times 1$ vector m is called the *mixing vector* and consists of a few non-zero entries associated to the images in the database that contribute to represent the query image by superposition. Note that the error e captures background clutter and the intra-class variability. A similar representation was introduced in [126] and was shown to be suitable for face recognition problems. We argue that is also a reasonable model for the generic object classification problem. As long as the training set is large

enough the image representation will yield satisfactorily discriminative features for classifying object classes as demonstrated in [49, 84]. In order to further justify the model, we show empirical evidence that mixing vectors m are both fairly sparse and concentrated using a number of datasets in the following Section 5.2.3.

5.2.3 Empirical evidence for sparse approximation

In this section, we provide empirical evidence of the assumption that a query image x can be both sparsely and accurately represented by a few linear combinations of BOW descriptors of the same category. The following experimental evaluation is carried out by using the hierarchical Caltech-256 dataset with 'dog' category. See Section 5.4 for more details about the structure of this dataset. Let us denote the $K \times C$ matrix H_s as the matrix that is formed by taking the columns in H that correspond to the same category as x. Thus, C is the number of images in a category. Note that, $K = 4200$ and $C = 30$ for this particular dataset and also that $K > C$. Then, we empirically show that $x = H_s m_s + e$ has a solution $\widehat{m}_s$ that is sparse and gives a small approximation error $\|x - H_s \widehat{m}_s\|_2$.

To compute $\widehat{m}_s$ for a given x we solve,

$$\min_{m_s} \|x - H_s m_s\|_2 + \lambda \|m_s\|_1,$$

which is also known as the least absolute shrinkage and selection operator (LASSO) [117]. The first term of the cost function ensures that the approximation error is small and the second term ensures that the solution is sparse. Figure 5.2 shows an example of a plot of $\widehat{m}_s$ obtained by solving the above minimization problem. We can see that $\widehat{m}_s$ is indeed sparse with only two non-zero coefficients and has a small approximation error of 0.58.

In order to demonstrate that such behavior is common across most queries x, we

repeat the above for a large number of queries x that belong to different categories and evaluate how sparse and accurate the solutions are by computing $\frac{\|\widehat{m}_s\|_1}{\|\widehat{m}_s\|_2}$ and $\|x - H_s\widehat{m}_s\|_2$, respectively. We note that $1 \leq \frac{\|m_s\|_1}{\|m_s\|_2} \leq \sqrt{30}$ and the closer this fraction is to 1 the sparser the $\widehat{m}_s$ and vice versa. The average value of $\frac{\|\widehat{m}_s\|_1}{\|\widehat{m}_s\|_2}$ and $\|x - H_s\widehat{m}_s\|_2$ for a large number of trials are 2.41 and 0.52, respectively, which show that x can indeed be sparsely and accurately represented by the columns of the same category.

Next we show that for a large number of queries, x is best represented by the columns of the same category than by those of other categories. In the Caltech-256 dataset there are in total 256 categories. For each query x, we solve the above minimization problem for all 256 categories, where each category has a different H_s that is constructed by taking the appropriate columns in H. Then for all 256 solutions, we evaluate $\frac{\|\widehat{m}_s\|_1}{\|\widehat{m}_s\|_2}$ and $\|x - H_s\widehat{m}_s\|_2$ as a measure of sparsity and accuracy. To asses whether or not x is better represented by the columns of the true category, we compute how many other categories resulted in a representation $\widehat{m}_s$ that gave 10% better performance in terms of the two measures simultaneously. We repeat this procedure for 512 different query images that belong to different categories and plot in Figure 5.2 the histogram of the number of categories that resulted in a better representation than the true category. Out of 512 trials for exactly 327 query images, the true category was able to better represent x than others. This and the fact that this histogram exhibits a high concentration close to zero shows that for most queries, the true category provides more sparse and accurate representations than other categories.

5.3 Classification

Clearly m contains the information that allows us to estimate the class label of the query image. Therefore, the classification problem (*what is the object class?*) is recast into the problem of estimating the vector m (*where is a non-zero entry?*). Furthermore, this formulation allows us to discover multiple dominant object categories

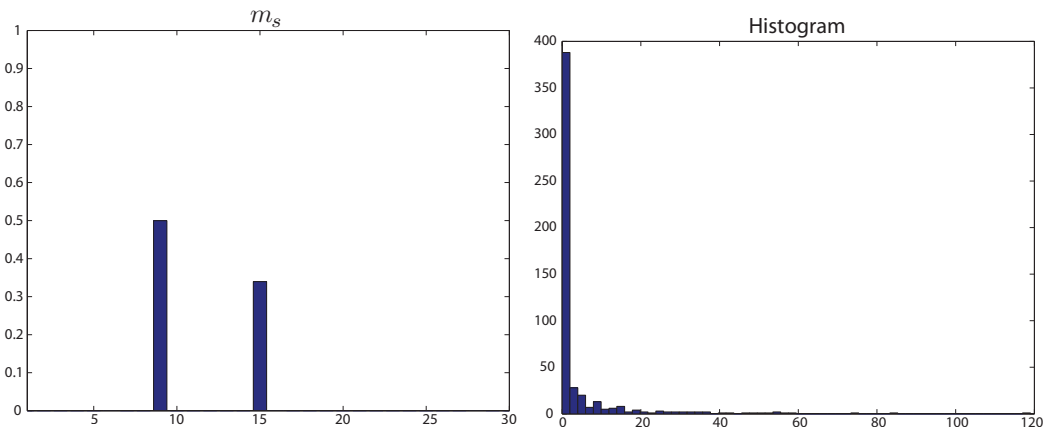


Figure 5.2: (left) An example of a reconstruction of $\widehat{m}_s$. In this example, $\widehat{m}_s$ has only two non-zero coefficients and $\|x - H_s\widehat{m}_s\|_2 = 0.58$. (right) Histogram of the number of categories that provided more sparse and accurate representations than the true category for 512 trials.

in the image. Suppose the image contains three objects as in Figure 5.1 (c), then the query image may be expressed as a superposition of $s = 3$ training histograms and the non-zero entries of m will return the 3 classes appearing in x (i.e, dog, human and vacuum). Solving m is challenging because the system is under-determined ($N \gg K$) and has an infinite number of solutions. Because we postulate or seek a s-sparse mixing vector m, we can formulate this problem as a sparse approximation problem and seek to find the sparsest solution that best approximates (in ℓ_0 error) the observed instance. Notice that the pseudo-norm $\|\cdot\|_0$ counts the number of non-zero entries in a vector.

Problem 0. $\min \|m\|_0 \quad \text{subject to} \quad \|Hm - x\|_2 \leq \epsilon.$

Unfortunately, the above problem is an NP-hard problem in general (given an arbitrary matrix H and an arbitrary vector x). We can, however, solve this problem in polynomial time with appropriate geometric assumptions on H; if the maximum entry of the matrix $|H^*H - I|$, or the coherence[1] $\mu(H)$, of the matrix is small, then

[1] An equivalent definition of $\mu(H)$ is the maximum dot-product of different columns of H, $\mu(H) = \max_{i \neq j} |\langle H_i, H_j \rangle|$.

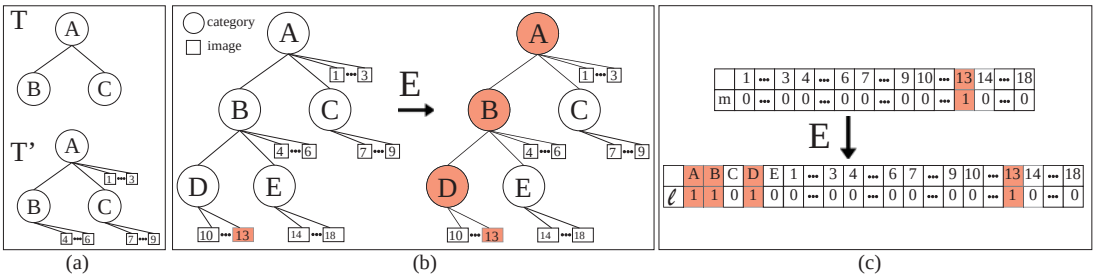


Figure 5.3: Visualization of the embedding. (a) Examples of $\mathcal{T}$ and $\mathcal{T}'$. (b,c) As a result of the embedding E, the flat mixing matrix m is mapped into l. In this example, when m shows a non-zero entry corresponding to image 13, the embedded l shows non-zero entries corresponding to image 13 as well as to its ancestors categories (nodes) A, B and D. These are on the path to the root from image 13.

there are several algorithmic solutions. Let us assume for now that the training set contains the query image x. As proposed by [18, 126], one method is to observe that Problem 0 is an optimization problem with a non-convex objective function and that a convex relaxation of this problem yields a problem which can be solved efficiently with standard optimization techniques [21],

Problem 1. $\min \|m\|_1 \quad \text{subject to} \quad \|Hm - x\|_2 \leq \epsilon.$

A second algorithmic approach is to use a greedy algorithm, one that identifies image instances iteratively, such as Orthogonal Matching Pursuit (OMP). See [120] and the references therein for details on this algorithm. In Section 5.4.3 we show that the coherence between individual images decreases as a function of their hierarchical distance; thus, while the overall coherence $\mu(H)$ is high, with high probability, the coherence between any two images is quite small and OMP can distinguish among these images and choose a representation close to the ground truth.

5.3.1 Hierarchical Classification with Sparse Approximation

While the model $x = Hm + e$ is reasonable and empirical evidence suggests that it is fairly accurate, it fails to take into account any hierarchical information

amongst the classes. Furthermore, the error metrics for typical sparse approximation algorithms [108, 120] do not take into account structural relationships amongst the columns of H. Indeed, a small error in the mixing vector $\|\widehat{m} - m\|_2$ or in the reconstruction of the observation x does not necessarily guarantee hierarchical similarity between $\widehat{m}$ and m. For instance, suppose the ground truth label of a query image is "dog". Assume two possible classification results are generated: "stapler" and "cat". These two results would be associated to the same *flat* classification error $\|\widehat{m} - m\|_2$ if the model in $x = Hm + e$ were employed, whereas the classification error associated to "cat" would be smaller than that associated to "stapler" if the error function were defined in hierarchical sense (Figure 5.1).

In this section, we assume that object categories are structured in a (rooted, labeled, recursive) tree $\mathcal{T}$ that reflects the semantic (parental) relationships among object categories. Note that each node of $\mathcal{T}$ contains all of the images representative of the visual category label associated to that node. A schematic illustration of such a data structure is given in Figure 5.1 and Figure 5.3. We define $\mathcal{T}'$, the data structure induced by the semantic tree, that contains two types of nodes, category labels and individual column vectors of H (images) (Figure 5.3). It encodes the semantic relationship amongst the categories and the assignment of columns of H to those categories, but, unlike the tree $\mathcal{T}$, both categories and individual columns of H make up the nodes. A key contribution of our work is to introduce a suitable encoding matrix E that embeds the flat model matrix H into a hierarchical (tree) model matrix Φ and to show that the resulting hierarchical sparse approximation is solvable and appropriate for classification.

5.3.2 Hierarchical embedding

The encoding matrix E is constructed so as to map the mixing vector m into an embedded mixing vector $\ell = Em$, whose non-zero entries correspond to the paths in

$\mathcal{T}'$ from the image to the root of the tree (Figure 5.3). More concretely, we define E to be the $(N + C) \times N$ matrix that embeds a column of H and its path to the root in the tree $\mathcal{T}'$. Without loss of generality, we can permute the rows of E so that E has the following structure $E = [I \quad L^T]^T$ where I is the $N \times N$ identity matrix and the $C \times N$ matrix L consists of the hierarchical labels of each image. Each row of L corresponds to a category and each column to a training image; $L_{i,j} = 1$ if category i is on the path to the root from training image j. Each row encodes which training images are descendants of category j. Note that the length of ℓ is $N+C$. If we denote $E^\dagger$ the pseudo-inverse of E, then we define $\Phi = HE^\dagger$.

5.3.3 Hierarchical sparse approximation

The hierarchical embedding allows to reformulate Problem 1 as a hierarchical sparse approximation problem and find a solution for ℓ given x:

Problem 2. $\min \|\ell\|_1 \quad \text{subject to} \quad \|\Phi\ell - x\|_2 \leq \epsilon$

Unlike the original sparse approximation problem, in this problem, the sparsity pattern of the vector ℓ is constrained to lie on a single path (or subtree) of the tree $\mathcal{T}'$. While the embedding $Em = \ell$ increases the number of non-zeros in ℓ (as compared to that of m), it also enforces a model that these non-zero entries must follow; they must lie on paths from individual columns of H to the root of the tree $\mathcal{T}'$. Because the sparsity of ℓ follows a model and Φ has more columns than rows, this problem has the structure of a model-based compressive sensing problem [4].

Problem 2 can be solved efficiently by a greedy algorithm called Tree-OMP [78], which is a special case of the more general algorithm Model-CoSaMP [4], assuming that Φ satisfies a geometric condition, referred to as model-Restricted Isometry Property (model RIP). (See Algorithm 2.) Tree-OMP is similar to the OMP algorithm with the additional step that for all non-zero components in the vector ℓ, the algorithm enforces that all the components that correspond to ancestors in the tree

Algorithm 2: TREE-OMP [78]

Input: Φ, x, ϵ, θ
Output: $\widehat{\ell}$
Initialize the the counter $k = 0$, the vector $\widehat{\ell}_k = 0$, the residual $r_k = x$, and the index set $\Lambda_k = \emptyset$.;
while *!done* **do**
  $z = \Phi^* r_k$;
  $S_k = \{\lambda \Big| |z_\lambda| > \theta\}$;
  // Form a candidate set of columns of Φ which have inner products with the residual larger than the threshold θ.
  $i_k = \arg\min_{i \in S_k} \|x - P_{span\{\phi_l : l \in F_i\}}\|_2$;
  // Search among the candidate set S_k for the item that together with its family F_i (i.e., all ancestor items) maximizes the reduction of residual.
  Set $\Lambda_{k+1} = \Lambda_k \cup F_{i_k}$.;
  $\ell_{k+1} = P_{span\{a_l : l \in \Lambda_{k+1}\}} x$;
  Update the residual, $r_{k+1} = x - \ell_{k+1}$.;
  if $\|r_{k+1}\|_2^2 < \epsilon$ **then** done;
  ;
  else $k = k + 1$.;
  ;
end

are non-zero. This constraint guarantees that the estimated solution $\widehat{\ell}$ corresponds to one (or more) physical path(s) in the tree.

5.3.4 Theoretical analysis

In this subsection, we show that the hierarchical embedding in Section 5.3.1 produces a matrix Φ that, on average, satisfies the model RIP. We also show that $\widehat{\ell}$, the output of TREE-OMP, is close to the ground truth embedded vector $\ell = Em$ not only in l_2 error, but, more importantly, in HD. These results are summarized in the following theorem. Moreover these results enable the construction of a classification algorithm that we call SPARSE PATH SELECTION (SPS) (see Figure 3).

Theorem 1. *Given a normalized test image x ($\|x\|_2 = 1$) which is sd-sparse with background "noise" n, we can solve $\Phi\ell = x + n$ for the embedded mixing vector ℓ*

Algorithm 3: Sparse Path Selection (SPS)

TRAINING
Input: Images with known hierarchy
Output: Encoded model matrix Φ, threshold value η

Form the matrix H of training vectors collected from all images in the dataset;
Encode: $\ell = Em$, $\Phi = HF$, where $F = E^{\dagger}$;
Normalize the columns of Φ to have unit l^2-norm.;
Learn the threshold value η;

TESTING
Input: Encoded model matrix Φ, query image, threshold value η
Output: Class labels

Form a vector x from the query image.;
Estimate mixing vector $\widehat{\ell}$ =TREE-OMP$(\Phi, x, \epsilon, \theta)$;
Truncate small elements in $\widehat{\ell}$ by learned threshold values η and return the labels of the remaining non-zero entries; i.e., the classification results.;

with TREE-OMP. *After* $T > \log(sd)$ *iterations, the output vector* $\widehat{\ell}$ *has at most* Td *non-zero entries and satisfies*

$$\|\ell - \widehat{\ell}\|_2 \leq 2^{-T} + C\|n\|_2.$$

In addition, if the noise $\|n\|_2 \leq \sqrt{Td}(\eta - 2^{-T})$ *is small enough compared to a learnt threshold* η *(See SPS algorithm), then* $HD(\widehat{\ell}, \ell) = 0$*; i.e., we correctly identify all the categories on the ground-truth hierarchical path.*

Proof. First, we note that the embedded vector $\ell = Em$ follows a model-sparse pattern as defined in [4].

Lemma 1. *If* m *is a* s*-sparse vector, then* $\ell = Em$ *has a sparse tree structure; that is, it encodes a rooted tree with* s *leaves.*

Proof. From [4], a signal model $\mathcal{M}_k$ is the union of m_k canonical k-dimensional subspaces $\mathcal{M}_k = \bigcup_{m=1}^{m_k} \chi_m$ where each k-dimensional subspace $\chi_m = \{y \big| y|_{\Omega_m^c} = 0\}$ contains all signals y with support in Ω_m. The model $\mathcal{M}_k$ is defined by the set of

possible k-sparse supports $\Omega_1, \ldots, \Omega_{m_k}$ and, if we restrict ourselves to those sets that are defined by a rooted tree structure, we have a model-sparse signal. Our embedding, by construction, yields such a vector ℓ; it is model $k \leq sd$ sparse (where d is the depth of the tree $\mathcal{T}'$). □

Lemma 2. *The matrix Φ is well-approximated by an iid (sub-)Gaussian random matrix.*

Proof. We model[2] the label matrix L as an iid random Bernoulli matrix; each entry $L_{i,j} = 1$ with probability p and 0 with probability $1 - p$. Let

$$\widetilde{E} = \frac{1}{2}\begin{bmatrix} I & \widetilde{L}^T \end{bmatrix}^T$$

where $\widetilde{L}_{i,j} = \frac{1}{Cp(1-p)}(L_{j,i} - p)$ is a centered version of the transpose of L. Observe that, on average, $\widetilde{E} = E^\dagger$, as

$$\mathbb{E}\Big((\widetilde{L}L)_{j,l}\Big) = \mathbb{E}\Big(\sum_{k=1}^{C} \widetilde{L}_{j,k}L_{k,l}\Big) = \sum_{k=1}^{C} \frac{1}{Cp(1-p)}\mathbb{E}\Big(L_{k,j} - p\Big)\mathbb{E}(L_{k,l}) = 0$$

and

$$\mathbb{E}\Big((\widetilde{L}L)_{j,j}\Big) = \mathbb{E}\Big(\frac{1}{Cp(1-p)}\sum_{k=1}^{C}(L_{k,j} - p)(L_{k,j})\Big) = \frac{1}{Cp(1-p)}\sum_{k=1}^{C} p(1-p) = 1.$$

Then, on average,

$$\Phi = H\widetilde{E} = \frac{1}{2}\begin{bmatrix} H & H\widetilde{L}^T \end{bmatrix}^T$$

[2]In practice, the assignment of labels to training images is deterministic and we have more descendant images for a category the higher in the tree it is. The indexing of the columns is, however, arbitrary so we can order them at random initially and fixed throughout the remainder of the algorithm. A more realistic model is to change the probability p as a function of the depth of the category in the tree. The root has $p = 1$ and a deep category has p close to 0.

and the entries in the columns corresponding to the $H\widetilde{L}$ block are

$$(H\widetilde{L})_{j,l} = \sum_{k=1}^{N} H_{j,k}\widetilde{L}_{k,l} = \sum_{k=1}^{N} H_{j,k}\frac{1}{Cp(1-p)}(L_{k,l} - p)$$

approximately iid Gaussian random variables as they are (large) sums of bounded random variables with mean 0.

This analysis describes the average behavior of Φ only. Any instance of $E^{\dagger}$ has non-zero entries in the off-diagonal terms. These entries are also bounded random variables, and, hence, the product $\Phi = HE^{\dagger}$ consists of entries that are approximately Gaussian random variables. □

From Lemma 1 and 2, we can conclude that Φ satisfies a model-RIP property [9]. Furthermore, we can use the result in [4] to conclude that after T iterations of Tree-OMP, the output $\widehat{\ell}$ contains at most Td non-zero entries and satisfies $\|\ell - \widehat{\ell}\|_2 \leq 2^{-T} + C\|n\|_2$. While the l_2 distance between two vectors is meaningful, it does not tell us how close the path(s) corresponding to the vector $\widehat{\ell}$ are compared to the ground-truth vector ℓ, it conflates the paths with the coefficients on those paths. The error bound tells us what the average error in $\widehat{\ell}$ is and, as long as it is below our learned threshold, $\frac{1}{\sqrt{Td}}(2^{-T} + \|n\|_2) < \eta$, we will not introduce spurious nodes in the path nor miss them and hence, $HD(\widehat{\ell}, \ell) = 0$. □

5.3.5 Sparse Path Selection Algorithm (SPS)

After solving Problem 2, we obtain an estimate of the path ℓ in the hierarchical database associated to the query image. However, ℓ cannot be used directly for image classification. Ideally, the sparsest solution of Problem 2 should return a vector of "1" and "0" where the non-zero elements in ℓ allows to estimate the category labels of the query object as well as its parents. Unfortunately, this is not always the case and values between "0" and "1" can be also found because of the estimation noise.

To solve this issue, we perform a post processing step. The idea is to introduce a threshold η and interpret it as a positive response any value that is above η (and as negative response, otherwise). Finding this threshold, however, is not trivial as it may vary with different datasets. Thus, in our experiments, we propose to automatically learn this thresholds using a binary MAP estimator trained using a validation set. Such evaluation set is then removed from the dataset so as to avoid contamination during testing. Our entire classification scheme is summarized in the Algorithm 3. We call this algorithm SPS.

5.3.6 Classifying multiple categories

As discussed in the previous sections, if the input vector x describes an image comprised of s categories, the mixing vector m is a s-sparse vector and the corresponding embedded mixing vector ℓ defines a subtree composed of s paths. Each of these paths is associated to one of the categories in x. (Figure 5.1) Thus, solving Problem 2 and obtaining an estimate $\widehat{m}$ of m allows us to *simultaneously* discover the presence of multiple categories in the image. Even if this appears to be an appealing property, one critical question must be addressed. How many categories s can we simultaneously handle until the conditions (i.e. sparsity, etc) underlying the solution of Problem 2 are violated? The bounds in [4] suggest that we need at least $O(sd)$ rows in the histograms, where d is depth of hierarchical tree and Section 5.4.7 gives some empirical evidence that multiple category classification is possible with these algorithms.

5.4 Experiments

In this section, we present quantitative and qualitative experimental results to validate our theoretical claims. We test our algorithm using different hierarchical databases. These are: i) 3 branches of the ImageNet [27] each comprising hundreds

of categories; ii) The hierarchical Caltech-256 dataset [48]. We use different metrics to evaluate the performances of our algorithm: i) Overall average Hamming Distance (HD); ii) Average classification accuracy for each levels of the hierarchy. We benchmark our results using competitive classification methods SRC, the sparse approximation technique introduced by [126]. Our experiments include classification of a single dominant object category as well as multiple categories. In each of the single category classification experiments we used 16 patches on a grid with step 8 pixels to generate SIFT descriptors. BOW histograms are constructed using 500 codewords generated from K-means clustering. Finally, we used SPH (Spatial Pyramid Histogram) up to the resolution level 4 to represent each image. In each experiment we sample (at most) 100 images for each node of the working database and use these for learning (i.e. to build the H matrix). For example, for the *domestic Animal* sub-tree of ImageNet, we collected about $21,000$ images for training. We sample an additional 10 images per node for testing. This way, testing images are guaranteed to be different from those in the training set.

5.4.1 ImageNet Subsets

ImageNet [27] is a hierarchical image database with 10 million images and over $10,000$ categories. It organizes different classes of images according to the WordNet [33] structure, and "IS-A" relationship exists between parents and children. In the experiments, we used 3 different branches from the ImageNet: *Home Appliances, Domestic Animals*, and *Fruits*. These subsets are chosen so as to observe the effect of different dataset sizes (48, 21, 320 respectively) and structures (domestic animal is a deeper tree than home-appliances) on the classification results.

5.4.2 Hierarchical Caltech-256

The Caltech-256 is rearranged in a hierarchy according to best matches in the WordNet. In this Hierarchical Caltech-256, all images are associated to a leaf node, hence there are no images in the internal nodes.

5.4.3 Dataset coherence properties

In this subsection, we analyze the coherence properties of different subsets of the datasets. We do not necessarily use all instances of every category but instead we pick two instances uniformly at random to obtain a statistical perspective on the coherence values of the derived H matrices.

Experimental results show that if we use all object instances, the matrix H is quite coherent and that the value of $\mu(H)$ is close to 1. A straightforward application of the previous theoretical results would suggest that neither the greedy algorithm nor the convex relaxation is appropriate for identifying a single instance of an object category. Notice that the case of multiple categories would be even more problematic. These theoretical results are, however, too pessimistic and do not explain all of our empirical observations[3]. We note that if we choose two objects independently and uniformly at random from the learned database, the coherence between these two objects decreases as a function of their distance in the hierarchical tree.

Figure 5.4 shows the relationship between the coherence of two objects (on average for objects chosen uniformly at random) and their distance (path length) in the hierarchical tree for the ImageNet dataset. This analysis suggests that the coherence measure is too fine a measure with which to evaluate our learned database; instances of the same object category are too similar while instances of different categories are, with high probability, dissimilar. Instead of tweaking the parameters of the "flat"

[3]In our experimental results, we can see that a sparse representation that does not take into account any structure amongst the instances is surprisingly successful, albeit far from the best solution.

sparse approximation in hopes of a small improvement, we should search for a sparse approximation that takes into account the hierarchical structure amongst the objects (and their categories).

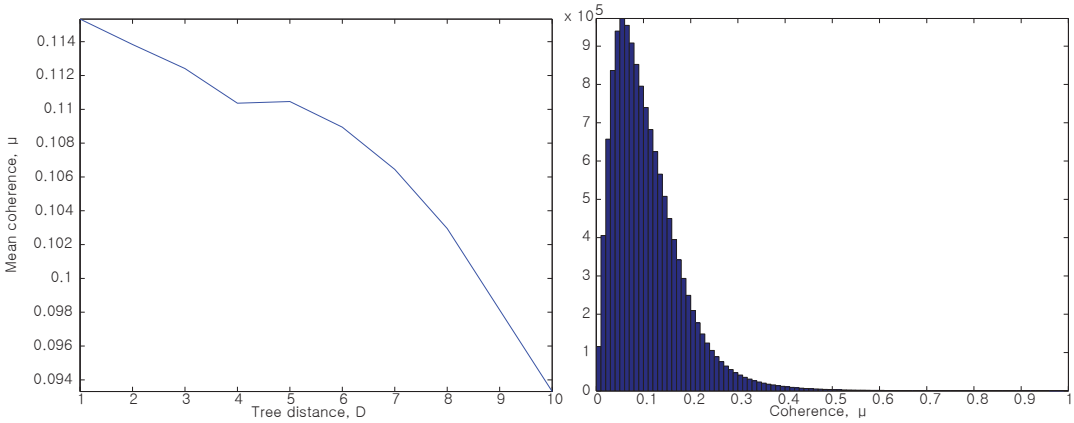


Figure 5.4: (left) Mean coherence for objects chosen uniformly at random as a function of their distance in the tree; (right) Distribution of coherence values for objects with distance 6 in the tree.

In practice, our data are not randomly generated. To test whether or not our data are consistent with our theoretical analysis, we show in Figure 5.5 a QQ-plot, which shows a simlilarity between two probablity distributions, for both a normal distribution and the entries of the matrices H (left) and Φ (right), respectively. Specifically, if samples are concentrated at diagonal lines, two distributions are similar. Thus, the plots show that the Φ distribution is closer to a normal distribution than H is but somewhat more skewed to negative values as compared to a normal distribution.

5.4.4 Benchmarks

The sparse approximation technique introduced by [126] (SRC) is used. We use Problem 1 (Section 5.2) to find the solution m via sparse approximation (similarly to [126]). We use the post-processing procedure in [126] to estimate the final class label. Notice that this method does not exploit the hierarchical structure of the

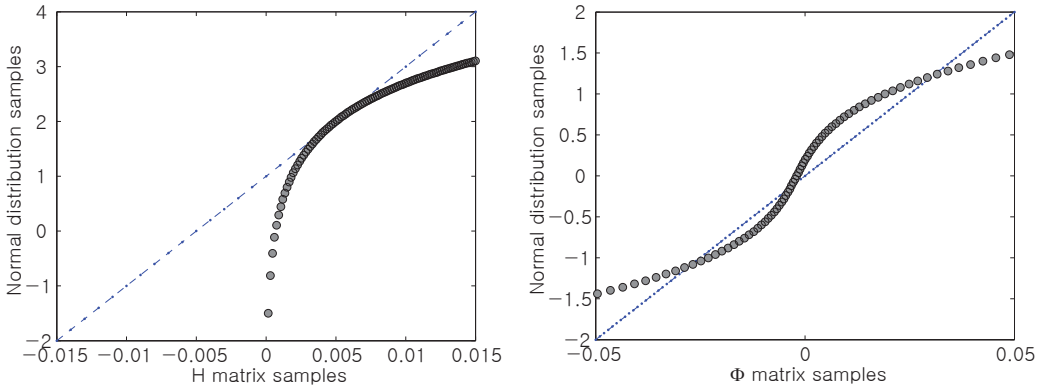


Figure 5.5: The QQ-plot can be used to verify that the matrix Φ obtained from embedding the Caltech-256 dataset is consistent with our theoretical observation that Φ is well-approximated by an iid random Gaussian matrix. The plot on the left is the QQ-plot for the original matrix H; the plot on the right is the QQ-plot for the matrix Φ. Observe that Φ is more consistent with a Gaussian random matrix than H is, although somewhat skewed compared to the normal.

database and "sees" the database as flat. Notice that SRC returns a single class label (not a path in the tree) which can be used to form the mixing vector m_{SRC}. In order to compare SRC results with ours, we embed m into its corresponding path $\ell_{\text{SRC}} = Em_{\text{SRC}}$. Notice that classifying ℓ correctly is as challenging as classifying m correctly since we don't know in advance the depth of the ground truth path.

5.4.5 Hierarchical Similarity Verification

In this section, we show classification results in terms of HD (which is a natural distance function to compare the similarity of two paths in a tree). Thus, if the ground truth path and the estimated path are similar, the HD will be small. In Figure 5.6 we show average HD between ground truth paths and estimated path for all our testing images using our approach (SPS). In the same figure we also report the HD distance between ground truth path and path estimated by SRC (i.e., ℓ_{SRC}). Note that the HD associated to our approach is systematically smaller for all the datasets. This result supports our argument that the proposed framework yields smaller HD bounds. Also, notice that when the hierarchical structure is relatively flat, the effect

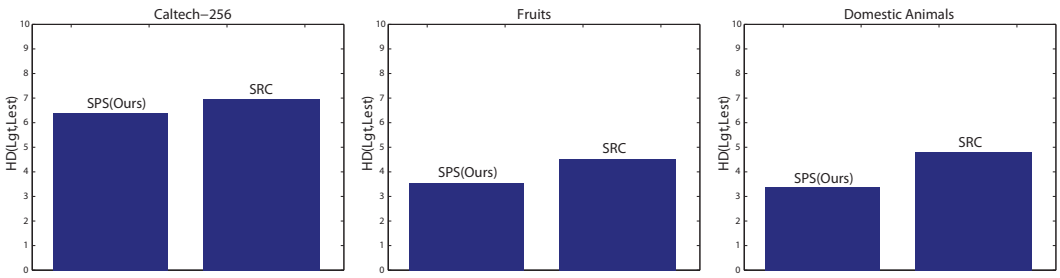


Figure 5.6: Average Hamming Distance (HD) for different subcategories is drawn.

of encoding (and thus the advantage from our framework) becomes less significant.

5.4.6 Effect on Different Hierarchy Levels

HD returns a global measurement of path similarity regardless of the level and position in the tree. In this experiment we explore the performance of our framework at different levels of the tree. Figure 5.7 plots the accuracy versus the levels of the hierarchy for different datasets (see caption for details). Notice that the root node is always classified correctly. As we go down toward the bottom of the tree, the likelihood of classifying nodes correctly becomes smaller and smaller. Also, note that this graph is always monotonically decreasing because whenever the estimation of the child category is correct, parent category estimation is correct too. When the hierarchical level is low, the performance of our SPS is similar to SRC. Interestingly, the plot shows that two algorithms yield equivalent performances in classifying images belonging to the leaf nodes. However, when the hierarchical level increases the gap between our SPS and SRC become much larger. This demonstrates the ability of our method to yield higher rates in classifying ancestors of the query object category. Anecdotal examples of paths returned by our SPS algorithm compared with those returned by SRC are shown in Figure 5.8. Note that estimated parent nodes returned by SRC are much less accurate than those returned by SPS. Paths are reported in text format.

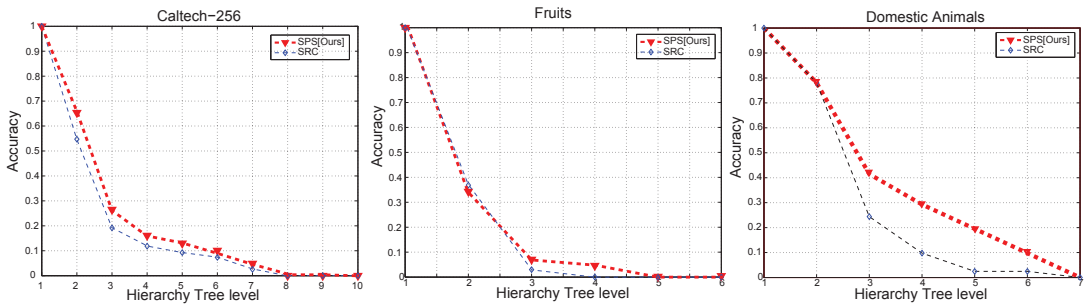


Figure 5.7: Average accuracy of classification for different hierarchical levels. We tested on five different categories, Caltech-256, Fruits, Domestic animals, Home applications and Domestic animals (leaves only). Average accuracy captures the average number of correctly estimated nodes (categories) for each level (x-axis) for all testing images. A node j is estimated correctly if the ground truth path evaluated at j is equal to the estimated path at j for a given test image. Consider the example in Figure 5.3 (b). In this case, the accuracy is calculated over 3 levels. Suppose the ground truth query image is 13 and the ground truth m is associated to class labels A, B, D. If the estimated m returns class labels A, B, E, the accuracy for three levels is $1, 1, 0$. If the estimated m returns the class labels A, C, the accuracy for three levels is $1, 0, 0$. If the estimated m just returns the class labels A, the accuracy for three levels is still $1, 0, 0$.

5.4.7 Multiple Category Classification

We report anecdotal examples demonstrating that our framework is able to classify images containing multiple categories. In such examples the histogram representing the query image can be expressed as a superimposition of multiple object category histograms. So, as discussed in the technical section, our SPS method will return *multiple* paths – a path for each of category in the query image. Examples in Figure 5.9 show some successful cases. Paths are reported in text format. The numerical results showing how accurately the algorithm is capable of retrieving multiple categories are shown in the Figure 5.10.

5.5 Conclusion

In this work, we introduced a novel framework for hierarchical classification using a new formulation of the sparse approximation problem. We demonstrated, for

the first time (up to our knowledge), that the hierarchical structure of a large and complex database can be indeed successfully used to enhance classification accuracy. Experimental results on several large scale dataset were used to support our claims.

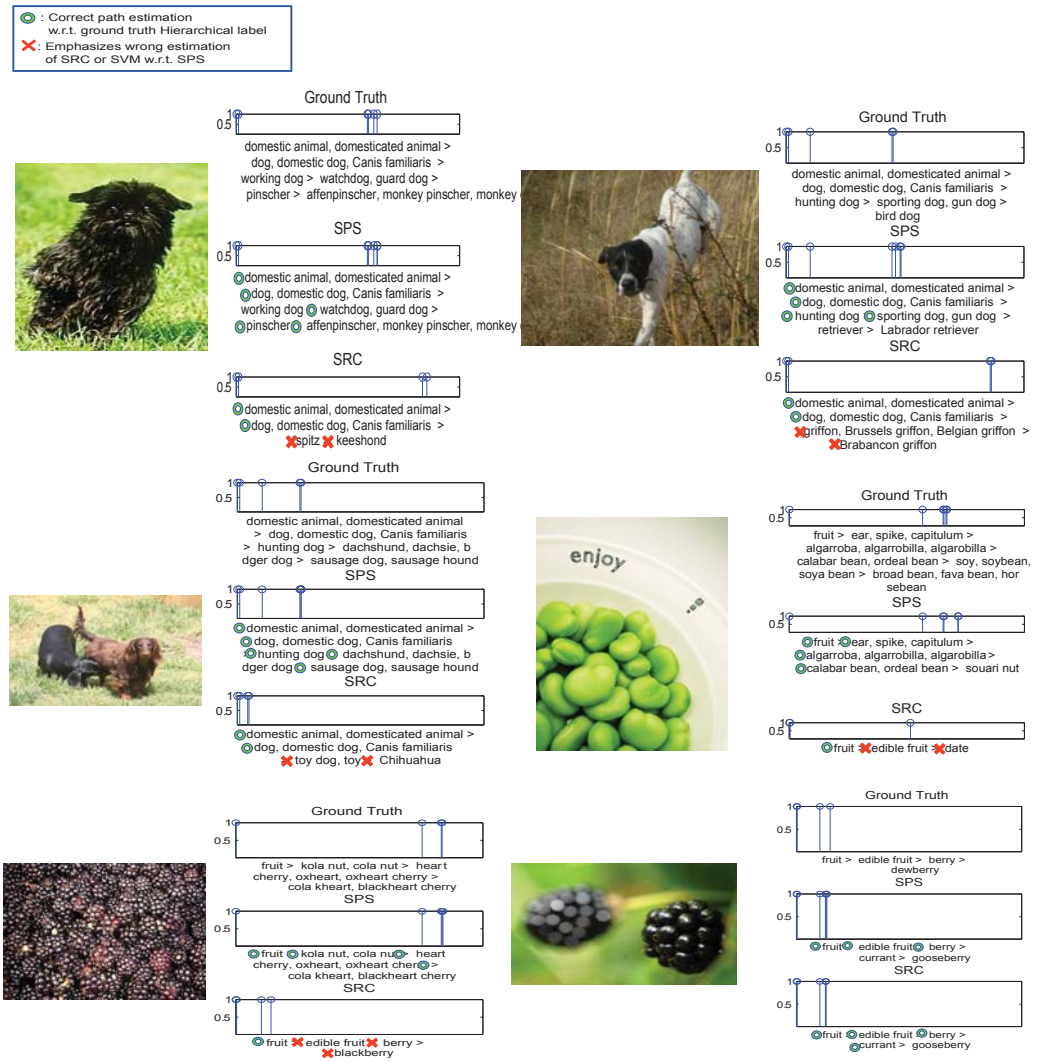


Figure 5.8: The hierarchical path is estimated as non-zero entries in the encoded mixing vector l. Note that the path estimated by SPS (ours) is closer to ground truth path than SRC is. Each response (non zero entry) corresponds to a classified category label for different levels of hierarchy. All these responses define the path l (=mixing matrix). For instance, look at the top left example. Ground truth path is domestic animal, domesticated animal > dog, domestic dog, Canis familiaris > working dog > watchdog, guard dog > pinscher > affenpinscher, monkey pinscher, monkey dog. Each category label, e.g., working dog corresponds to a non-zero entry in the mixing matrix l. working dog is a category within the third level of the hierarchy. Notice that the SPS (our algorithm) returns the correct path (each category for each level is correctly estimated). Conversely, SRC estimates the first two levels correctly, but it returns the wrong estimation starting from the third level.

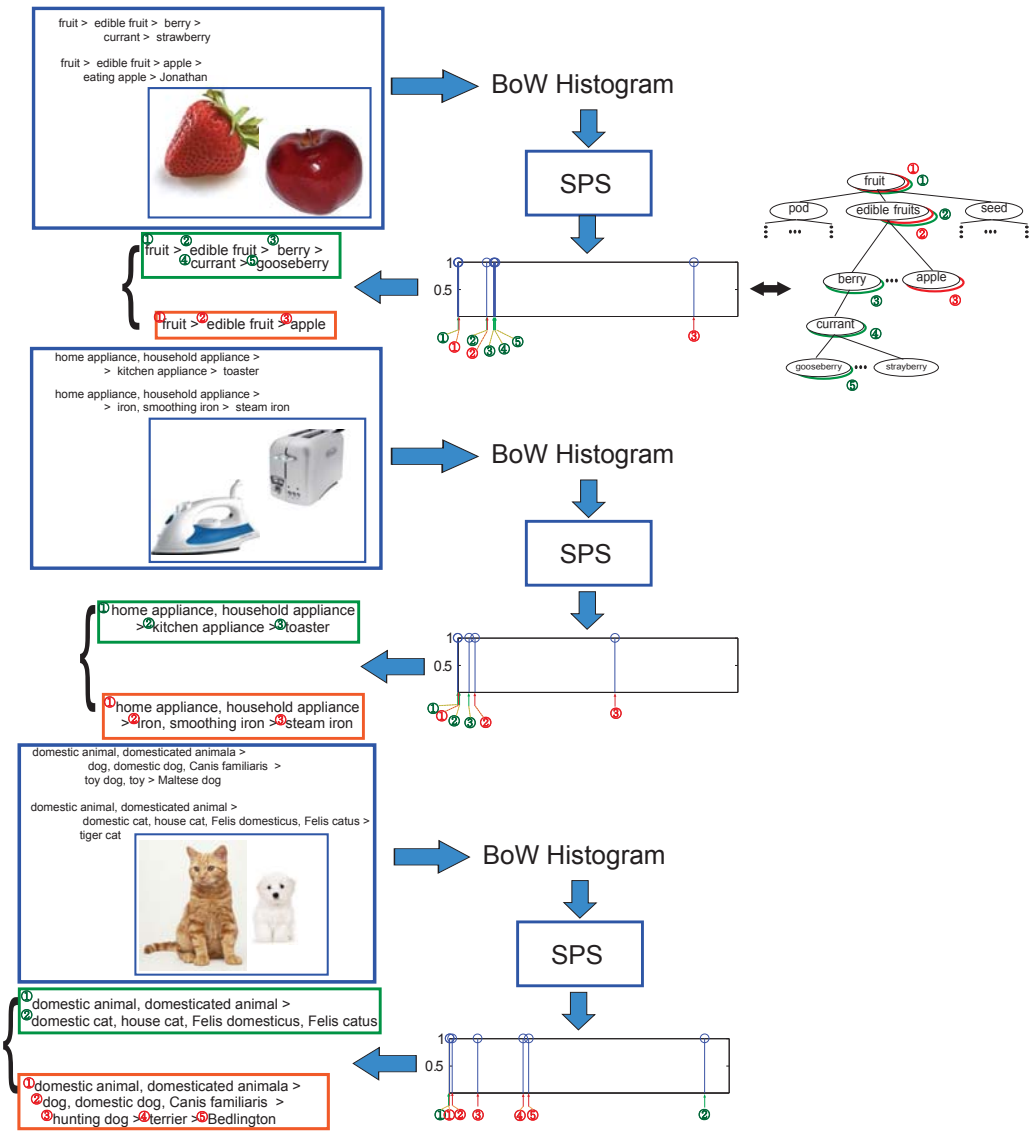
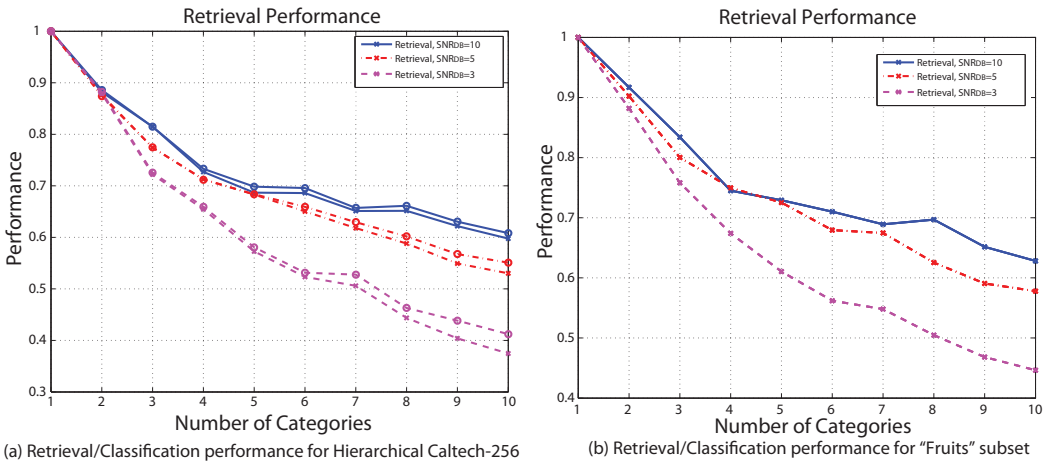


Figure 5.9: Multi objects recognition examples. Each test image contains two object categories. A level-zero pyramid histogram of codewords is used to represent the image in this case. Our SPS algorithm returns a mixing matrix where two paths can be identified. Each path is associated to a different object category. In top example, the estimated path associated to apple (i.e., fruit > edible fruit > apple) is indicated in red; the estimated path associated to gooseberry (i.e. fruit > edible fruit > berry > currant > gooseberry) is indicated in green.



(a) Retrieval/Classification performance for Hierarchical Caltech-256

(b) Retrieval/Classification performance for "Fruits" subset

Figure 5.10: Numerical results showing how accurately the algorithm is capable to retrieve multiple images at the same time (in this case we assume query images are also contained in the database; we point out that in all the other classification experiments presented in this work, test images are not contained in the database). Here x is a superposition of (up to) 10 histograms, and the goal is retrieve the ground truth paths given x. In this experiment, in order to simulate the effect of background clutter and intra-class variability, we added Gaussian noise on top of query image so as to have SNR_{dB} from 3 to 10. As expected retrieval performances decrease as the number of categories increases, or the noise ratio increases. This analysis is interesting as it can be related with our theoretical findings.

CHAPTER VI

Conclusion

In this thesis, we have proposed a number of models for understanding images by recognizing different components of the image (*things*, *stuff*, and *geometry*) in a joint fashion. Unlike current approaches that recognize each of these components independently, the proposed models incorporate relationships between components as well as their individual properties. This allows us to achieve state-of-the-art performances in many challenging computer vision problems. Here we summarize the contributions and discuss future work.

First, in Chap. II, we proposed a framework that recognizes *things* and *stuff* in a joint fashion. Various types of co-occurence between *things* and *stuff* have been utilized for scene understanding, and we have shown experimental results that support our model. Second, in Chap. III, we have proposed the model for understanding a 3D scene from a RGB-D image. With the proposed voxel-based CRF (V-CRF) model, we estimate not only the semantic labels of a 3D space, but also the occupancy of the 3D space. By having 3D priors for planar surface region, we recover geometric and semantic properties of occluded regions. Experimental results show that V-CRF model successfully understand a 3D scene, and the model has the potential to be used for 3D applications such as augmented reality. Third, a method for accurate 3D object localization using a RGB-D image is introduced in Chap. IV. By training the

discriminative model with 2D/3D features extracted from multiple foreground object masks, it was able to accurately localize the object in 3D space; furthermore, the 2D object localization performance has been improved. Finally, in Chap. V, we have proposed the model for image classification from a hierarchically organized dataset. The model finds sparse paths in a hierarchical structure that corresponds to a query image by matching features from candidate paths with features from a testing image. Compared to an image classification method based on images without structures, it achieves significant improvement in classification accuracy. In addition, the model was able to identify more than two objects in a single image.

A number of possible future works is considered. First, we plan to incorporate accurate 3D priors such as 3D CAD models for scene understanding. This will allow us to recover details when a subset of objects is occluded or depth data is contaminated. Second, we plan to use more flexible representation such as meshes or octrees on top of voxels for scene understanding in a finer resolution. Third, we are interested in real-time 3D scene understanding where one must manage a difficult balance between complexity and computational cost.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Alpert, S., M. Galun, R. Basri, and A. Brandt (2007), Image segmentation by probabilistic bottom-up aggregation and cue integration, in *CVPR*.

[2] Bao, S. Y., and S. Savarese (2011), Semantic structure from motion, in *CVPR*.

[3] Bao, S. Y., M. Sun, and S. Savarese (2010), Toward coherent object detection and scene layout understanding, in *CVPR*.

[4] Baraniuk, R., V. Cevher, M. Duarte, and C. Hegde (2008), Model-based compressive sensing, *preprint*.

[5] Barinova, O., V. Lempitsky, and P. Kohli (2012), On detection of multiple object instances using hough transforms, *PAMI*.

[6] Bay, H., T. Tuytelaars, and L. Van Gool (2006), Surf: Speeded up robust features, in *Computer Vision–ECCV 2006*, pp. 404–417, Springer.

[7] Binder, A., M. Kawanabe, and U. Brefeld (2003), Efficient Classification of Images with Taxonomies.

[8] Bleyer, M., C. Rother, P. Kohli, D. Scharstein, and S. Sinha (2011), Object stereo: joint stereo matching and object segmentation, in *CVPR*.

[9] Blumensath, T., and M. Davies (2007), Sampling theorems for signals from the union of linear subspaces, *IEEE Trans. Info. Theory*, pp. 30–56.

[10] Bo, L., K. Lai, X. Ren, and D. Fox (2011), Object recognition with hierarchical kernel descriptors, in *ICCV*.

[11] Borenstein, E., and S. Ullman (2008), Combined top-down/bottom-up segmentation, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *30*(12), 2109–2125.

[12] Boros, E., and P. Hammer (2002), Pseudo-boolean optimization., *Discrete Applied Mathematics*.

[13] Borrmann, D., J. Elseberg, K. Lingemann, and A. Nüchter (2011), The 3d hough transform for plane detection in point clouds: A review and a new accumulator design, *3D Research*.

[14] Bosch, A., A. Zisserman, and X. Munoz (2006), Scene classification via plsa, *Computer Vision–ECCV 2006*, pp. 517–530.

[15] Bosch, A., A. Zisserman, and X. Muoz (2008), Scene classification using a hybrid generative/discriminative approach, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *30*(4), 712–727.

[16] Bosch, X. B., J. M. Gonfaus, J. van de Weijer, A. D. Bagdanov, J. Serrat, and J. Gonz'alez (2010), Harmony potentials for joint classification and segmentation, in *CVPR*.

[17] Boykov, Y., O. Veksler, and R. Zabih (2001), Fast approximate energy minimization via graph cuts, *PAMI*.

[18] Candès, E., J. Romberg, and T. Tao (2006), Stable signal recovery from incomplete and inaccurate measurements, *Communications on Pure and Applied Mathematics*, *59*(8), 1207.

[19] Cao, L., and L. Fei-Fei (2007), Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes, in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, IEEE.

[20] Carreira, J., and C. Sminchisescu (2010), Constrained parametric min-cuts for automatic object segmentation, in *CVPR*, pp. 3241–3248.

[21] Chen, S., D. Donoho, and M. Saunders (2001), Atomic decomposition by basis pursuit, *SIAM review*, *43*(1), 129–159.

[22] Choi, W., Y.-W. Chao, C. Pantofaru, and S. Savarese (2013), Understanding indoor scenes using 3d geometric phrases, in *CVPR*.

[23] Csurka, G., C. Dance, L. Fan, J. Willamowski, and C. Bray (2004), Visual categorization with bags of keypoints, in *Workshop on Statistical Learning in Computer Vision, ECCV*, vol. 1, p. 22, Citeseer.

[24] Dalal, N., and B. Triggs (2005), Histograms of oriented gradients for human detection, in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE.

[25] Darom, T., and Y. Keller (2012), Scale-invariant features for 3-d mesh models, *Image Processing*, *21*(5), 2758–2769.

[26] Davison, A. J. (2003), Real-time simultaneous localisation and mapping with a single camera, in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1403–1410, IEEE.

[27] Deng, J., W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei (2009), ImageNet: a large-scale hierarchical image database, in *Proc. CVPR*, pp. 710–719.

[28] Desai, C., D. Ramanan, and C. Fowlkes (2009), Discriminative models for multi-class object layout, in *ICCV*.

[29] Dissanayake, M. G., P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba (2001), A solution to the simultaneous localization and map building (slam) problem, *Robotics and Automation, IEEE Transactions on*, *17*(3), 229–241.

[30] Everingham, M., A. Zisserman, C. Williams, and L. Van Gool (2006), The PASCAL visual object classes challenge 2006 (VOC2006) results, in *Workshop in ECCV06, May. Graz, Austria*, Citeseer.

[31] Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2009), The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results.

[32] Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin (2008), Liblinear: A library for large linear classification, *The Journal of Machine Learning Research*, *9*, 1871–1874.

[33] Fellbaum, C., et al. (1998), *WordNet: An electronic lexical database*, MIT press Cambridge, MA.

[34] Felzenszwalb, P., and D. Huttenlocher (2004), Distance transforms of sampled functions.

[35] Felzenszwalb, P., D. McAllester, and D. Ramanan (2008), A discriminatively trained, multiscale, deformable part model, in *CVPR*.

[36] Felzenszwalb, P., R. Girshick, D. McAllester, and D. Ramanan (2010), Object detection with discriminatively trained part-based models, *TPAMI*, *32*(9), 1627–1645.

[37] Felzenszwalb, P. F., and D. P. Huttenlocher (2005), Pictorial structures for object recognition, *IJCV*.

[38] Felzenszwalb, P. F., R. B. Girshick, and D. McAllester (2010), Discriminatively trained deformable part models, release 4, http://people.cs.uchicago.edu/ pff/latent-release4/.

[39] Forsyth, D. A., J. Malik, M. M. Fleck, H. Greenspan, T. Leung, S. Belongie, C. Carson, and C. Bregler (1996), *Finding pictures of objects in large collections of images*, Springer.

[40] Fritz, M., K. Saenko, and T. Darrell (2010), Size matters: Metric visual search constraints from monocular metadata, *NIPS*.

[41] Gall, J., and V. Lempitsky (2009), Class-specific hough forests for object detection, in *CVPR*.

[42] Geiger, A., P. Lenz, and R. Urtasun (2012), Are we ready for autonomous driving? the kitti vision benchmark suite, in *CVPR*.

[43] Gonfaus, J., X. Boix, J. Van De Weijer, A. Bagdanov, J. Serrat, and J. Gonzalez (2010), Harmony potentials for joint classification and segmentation, in *CVPR*.

[44] Gould, S., R. Fulton, and D. Koller (2009), Decomposing a scene into geometric and semantically consistent regions, in *ICCV*.

[45] Gould, S., T. Gao, and D. Koller (2009), Region-based segmentation and object detection, in *NIPS*.

[46] Gould, S., O. Russakovsky, I. Goodfellow, P. Baumstarck, A. Ng, and D. Koller (2009), The stair vision library (v2.3).

[47] Grauman, K., and T. Darrell (2005), The pyramid match kernel: discriminative classification with sets of image features, in *ICCV*.

[48] Griffin, G., and P. Perona (2008), Learning and using taxonomies for fast visual categorization, in *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pp. 1–8.

[49] Griffin, G., A. Holub, and P. Perona (2007), Caltech-256 object category dataset.

[50] Guivant, J., E. Nebot, and H. Durrant-Whyte (2000), Simultaneous localization and map building using natural features in outdoor environments, in *Intelligent Autonomous Systems*, vol. 6, pp. 581–586.

[51] Guo, R., and D. Hoiem (2012), Beyond the line of sight: labeling the underlying surfaces, in *ECCV*.

[52] Gupta, A., and L. Davis (2008), Beyond nouns: Exploiting prepositions and comparators for learning visual classifiers, in *ECCV*.

[53] Gupta, A., A. Efros, and M. Hebert (2010), Blocks world revisited: Image understanding using qualitative geometry and mechanics, *ECCV*.

[54] Gupta, A., A. Efros, and M. Hebert (2010), Blocks world revisited: Image understanding using qualitative geometry and mechanics, *ECCV*.

[55] Gupta, A., S. Satkin, A. A. Efros, and M. Hebert (2011), From 3d scene geometry to human workspace, in *CVPR*.

[56] Gupta, S., P. Arbelaez, and J. Malik (2013), Perceptual organization and recognition of indoor scenes from rgb-d images, in *CVPR*.

[57] Han, F., and S.-C. Zhu (2009), Bottom-up/top-down image parsing with attribute grammar, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *31*(1), 59–73.

[58] He, X., R. S. Zemel, and M. Á. Carreira-Perpiñán (2004), Multiscale conditional random fields for image labeling, in *CVPR*.

[59] Hedau, V., D. Hoiem, and D. Forsyth (2009), Recovering the spatial layout of cluttered rooms, in *Computer vision, 2009 IEEE 12th international conference on*, pp. 1849–1856, IEEE.

[60] Hedau, V., D. Hoiem, and D. Forsyth (2010), Thinking inside the box: Using appearance models and context based on room geometry, in *Computer Vision–ECCV 2010*, pp. 224–237, Springer.

[61] Hedau, V., D. Hoiem, and D. Forsyth (2012), Recovering free space of indoor scenes from a single image, in *CVPR*.

[62] Heitz, G., and D. Koller (2008), Learning spatial context: Using stuff to find things, in *ECCV*.

[63] Heitz, G., S. Gould, A. Saxena, and D. Koller (2008), Cascaded classification models: Combining models for holistic scene understanding, in *NIPS*.

[64] Hoiem, D., A. A. Efros, and M. Hebert (2005), Geometric context from a single image, in *ICCV*.

[65] Hoiem, D., A. A. Efros, and M. Hebert (2006), Putting objects in perspective, in *CVPR*.

[66] Hoiem, D., A. A. Efros, and M. Hebert (2008), Closing the loop on scene interpretation, in *CVPR*.

[67] Janoch, A., S. Karayev, Y. Jia, J. Barron, M. Fritz, K. Saenko, and T. Darrell (2011), A category-level 3-d object dataset: Putting the kinect to work, in *ICCV Workshops*, pp. 1168–1174.

[68] Jia, Z., A. Gallagher, A. Saxena, and T. Chen (2013), 3d-based reasoning with blocks, support, and stability, in *CVPR*.

[69] Joachims, T., T. Finley, and C. Yu (2009), Cutting-plane training of structural svms, *Machine Learning*.

[70] Kim, B., M. Sun, P. Kohli, and S. Savarese (2012), Relating things and stuff by high-order potential modeling, in *ECCV Workshop (HiPot)*.

[71] Knopp, J., M. Prasad, G. Willems, R. Timofte, and L. Van Gool (2010), Hough transform and 3d surf for robust three dimensional classification, in *ECCV*.

[72] Kohli, P., L. Ladicky, and P. H. S. Torr (2008), Robust higher order potentials for enforcing label consistency, in *CVPR*.

[73] Kolmogorov, V., and R. Zabih (2004), What energy functions can be minimized via graph cuts., *PAMI*.

[74] Koppula, H., A. Anand, T. Joachims, and A. Saxena (2011), Semantic labeling of 3d point clouds for indoor scenes, in *NIPS*.

[75] Koppula, H. S., A. Anand, T. Joachims, and A. Saxena (2011), Semantic labeling of 3d point clouds for indoor scenes, in *NIPS*.

[76] Krähenbühl, P., and V. Koltun (2012), Efficient inference in fully connected crfs with gaussian edge potentials, *arXiv preprint arXiv:1210.5644*.

[77] Kumar, S., and M. Hebert (2003), Discriminative fields for modeling spatial dependencies in natural images, NIPS.

[78] La, C., and M. Do (), Tree-based orthogonal matching pursuit algorithm for signal reconstruction, in *IEEE International Conference on Image Processing (ICIP)*, pp. 1277–1280, Citeseer.

[79] Ladicky, L., C. Russell, P. Kohli, and P. Torr (2009), Associative hierarchical crfs for object class image segmentation, in *ICCV*.

[80] Ladicky, L., C. Russell, P. Kohli, and P. H. Torr (2010), Graph cut based inference with co-occurrence statistics, in *ECCV*.

[81] Ladickỳ, L., P. Sturgess, K. Alahari, C. Russell, and P. Torr (2010), What, where and how many? combining object detectors and crfs, *ECCV*.

[82] Ladickỳ, L., P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. Torr (2012), Joint optimization for object class segmentation and dense stereo reconstruction, *IJCV*.

[83] Lai, K., L. Bo, X. Ren, and D. Fox (2011), A large-scale hierarchical multi-view rgb-d object dataset, in *ICRA*, pp. 1817–1824.

[84] Lazebnik, S., C. Schmid, and J. Ponce (2006), Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, Citeseer.

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

[86] Lee, H., R. Grosse, R. Ranganath, and A. Y. Ng (2009), Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 609–616, ACM.

[87] Leibe, B., A. Leonardis, and B. Schiele (2004), Combined object categorization and segmentation with an implicit shape model, in *ECCV workshop on statistical learning in computer vision*.

[88] Li, L.-J., R. Socher, and L. Fei-Fei (2009), Towards total scene understanding:classification, annotation and segmentation in an automatic framework, in *CVPR*.

[89] Lowe, D. G. (2004), Distinctive image features from scale-invariant keypoints, *International journal of computer vision*, *60*(2), 91–110.

[90] Mairal, J., et al. (2009), Online learning for matrix factorization and sparse coding, *stat*, *1050*, 1.

[91] Maire, M., S. Yu, and P. Perona (2011), Object detection and segmentation from joint embedding of parts and pixels, in *ICCV*.

[92] Maji, S., and J. Malik (2009), Object detection using a max-margin hough tranform, in *CVPR*.

[93] Marszalek, M., and C. Schmid (2007), Semantic hierarchies for visual object recognition.

[94] Microsoft (), Microsoft kinect, http://www.xbox.com/en-us/kinect.

[95] Munoz, D., J. A. Bagnell, and M. Hebert (2010), Stacked hierarchical labeling, in *ECCV*.

[96] Munoz, D., J. A. Bagnell, and M. Hebert (2012), Co-inference machines for multi-modal scene analysis, in *European Conference on Computer Vision (ECCV)*.

[97] Nathan Silberman, P. K., Derek Hoiem, and R. Fergus (2012), Indoor segmentation and support inference from rgbd images, in *ECCV*.

[98] Palmer, S. (1999), *Vision science: Photons to phenomenology*, MIT press Cambridge, MA.

[99] Pepik, B., M. Stark, P. Gehler, and B. Schiele (2012), Teaching 3d geometry to deformable part models, in *CVPR*.

[100] Perina, A., M. Cristani, U. Castellani, V. Murino, and N. Jojic (2009), A hybrid generative/discriminative classification framework based on free-energy terms, in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2058–2065, IEEE.

[101] Project page, http://cvgl.stanford.edu/projects/vcrf/ ().

[102] Rabinovich, A., A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie (2007), Objects in context, in *ICCV*.

[103] Ren, X., L. Bo, and D. Fox (2012), Rgb-(d) scene labeling: Features and algorithms, in *CVPR*.

[104] Rother, C., V. Kolmogorov, V. Lempitsky, and M. Szummer (2007), Optimizing binary mrfs via extended roof duality, in *CVPR*.

[105] Sadeghi, M., and A. Farhadi (2011), Recognition using visual phrases, in *CVPR*.

[106] Salas-Moreno, R. F., R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison (2013), Slam++: Simultaneous localisation and mapping at the level of objects, in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 1352–1359, IEEE.

[107] Satkin, S., J. Lin, and M. Hebert (2012), Data-driven scene understanding from 3d models.

[108] Saunders, S. Chen, and D. Donoho (1996), Atomic decomposition by basis pursuit, *SIAM J. Sci. Comput*.

[109] Shotton, J., J. Winn, C. Rother, and A. Criminisi (2006), Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation, in *Computer Vision–ECCV 2006*, pp. 1–15, Springer.

[110] Shotton, J., A. Blake, and R. Cipolla. (2008), Semantic texton forests for image categorization and segmentation., in *CVPR*.

[111] Silberman, N., and R. Fergus (2011), Indoor scene segmentation using a structured light sensor, in *Proceedings of the International Conference on Computer Vision - Workshop on 3D Representation and Recognition*.

[112] Silberman, N., D. Hoiem, P. Kohli, and R. Fergus (2012), Indoor segmentation and support inference from rgbd images, ECCV.

[113] Steder, B., R. Rusu, K. Konolige, and W. Burgard (2011), Point feature extraction on 3d range scans taking into account object boundaries, in *ICRA*.

[114] Sun, M., H. Su, S. Savarese, and L. Fei-Fei (2009), A multi-view probabilistic model for 3d object classes, in *CVPR*.

[115] Sun, M., S. Y. Bao, and S. Savarese (2010), Object detection with geometrical context feedback loop, in *BMVC*.

[116] Sun, M., G. Bradski, B.-X. Xu, and S. Savarese (2010), Depth-encoded hough voting for coherent object detection, pose estimation, and shape recovery, in *ECCV*.

[117] Tibshirani, R. (1994), Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society, Series B*, *58*, 267–288.

[118] Tighe, J., and S. Lazebnik (2010), Superparsing: Scalable nonparametric image parsing with superpixels, in *ECCV*.

[119] Tola, E., V. Lepetit, and P. Fua (2010), Daisy: An efficient dense descriptor applied to wide-baseline stereo, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *32*(5), 815–830.

[120] Tropp, J. (2004), Greed is good: Algorithmic results for sparse approximation, *IEEE Transactions on Information Theory*, *50*(10), 2231–2242.

[121] Tsochantaridis, I., T. Hofmann, T. Joachims, and Y. Altun (2004), Support vector learning for interdependent and structured output spaces, in *ICML*.

[122] Van Krevelen, D., and R. Poelman (2010), A survey of augmented reality technologies, applications and limitations, *International Journal of Virtual Reality*.

[123] Vedaldi, A., V. Gulshan, M. Varma, and A. Zisserman (2009), Multiple kernels for object detection, in *ICCV*.

[124] Wang, J., J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong (2010), Locality-constrained linear coding for image classification, in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3360–3367, IEEE.

[125] Winn, J., and J. Shotton (2006), The layout consistent random field for recognizing and segmenting partially occluded objects, in *CVPR*.

[126] Wright, J., A. Yang, A. Ganesh, S. Sastry, and Y. Ma (2009), Robust face recognition via sparse representation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 210–227.

[127] Xiang, Y., and S. Savarese (2012), Estimating the aspect layout of object categories, in *CVPR*.

[128] Yang, J., K. Yu, Y. Gong, and T. Huang (2009), Linear spatial pyramid matching using sparse coding for image classification, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1794–1801, IEEE.

[129] Yu, C.-N. J., and T. Joachims (2009), Learning structural svms with latent variables, in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1169–1176, ACM.

[130] Zweig, A., and D. Weinshall (2007), Exploiting object hierarchy: Combining models from different category levels, in *IEEE 11th International Conference on Computer Vision*, pp. 1–8.